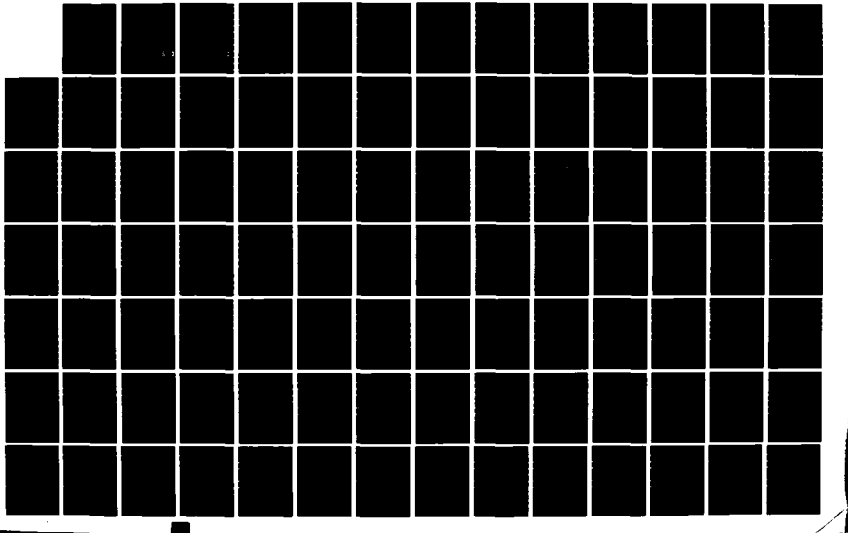
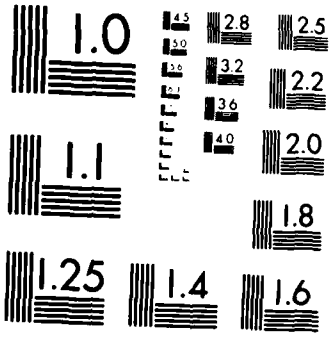
AD-A151 690 LOW-RESOLUTION TARGET CLASSIFICATION FROM A STARING INFRARED SENSOR(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING 1/2

UNCLASSIFIED J N TILLEY DEC 84 AFIT/GE/ENG/84D-67 F/G 17/5 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963 A

REPRODUCED AT GOVERNMENT EXPENSE

AD-A151 690

LOW-RESOLUTION TARGET CLASSIFICATION FROM A STARING INFRARED SENSOR

THESIS

James N. Tilley, III, B.S.E.E.
First Lieutenant, USAF

AFIT/GE/ENG/84D-67

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

DTIC
ELECTE
MAR 28 1985
S D
B

DTIC FILE COPY

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

85 03 13 152

AFIT/GE/ENG/84D-67

LOW-RESOLUTION TARGET CLASSIFICATION FROM A STARING INFRARED SENSOR

THESIS

James N. Tilley, III, B.S.E.E.
First Lieutenant, USAF

AFIT/GE/ENG/84D-67

DTIC
ELECTE
MAR 2 8 1985
S D
B

Approved for public release; distribution unlimited

AFIT/GE/ENG/84D-67

LOW-RESOLUTION TARGET CLASSIFICATION FROM A STARING INFRARED SENSOR

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Electrical Engineering

James N. Tilley III, B.S.
First Lieutenant, USAF

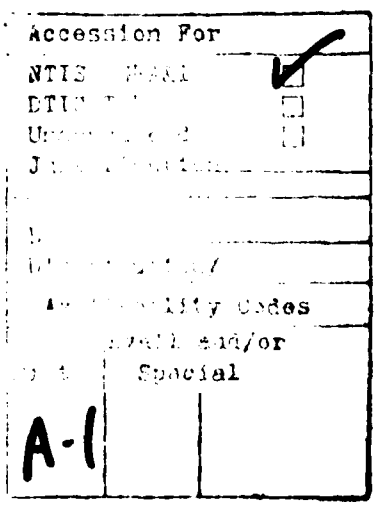
December 1984

Approved for public release; distribution unlimited

Preface

For those who have had the opportunity to view real staring infrared data may have observed instances when a target blob (aircraft) moving across the focal plane seemed to possess "target-like" qualities. These glimpses of target characteristics may not be obvious in single frames of data, but can be inferred from the motion of the target. The purpose of this study is to develop processing methods that exploit the dynamic nature of the target by recovering enough information to identify the target. Unfortunately, actual staring sensor data was not available to test the algorithms developed here.

I would like to thank my thesis advisor, Dr. Mathew Kabrisky, and Captain David King who provided invaluable guidance essential to the success of this study. Also, I am grateful to my sponsor, Foreign Technology Division/SQDA, for providing timely assistance in obtaining data and reference material. I also wish to acknowledge Dr. Thomas Janssens of the Aerospace Corporation who provided guidance in the early stages of this study. Finally, I would like to thank my wife, Diane, for her patience and constant encouragement.



Contents

List of Figures

List of Tables

List of Symbols

A_e	-	Area of Aircraft Engine (meter^2)
A_M	-	Area of Radiant Background Masked by Target (varies from 0 to A_T meters^2)
A_o	-	Effective Area Optics Aperture
A_T	-	Area of Target In Detector Field-of-View (meters^2)
c	-	Speed of Light
DFOV	-	Detector Field-of-View
E_{BKG}	-	The Average Irradiance From Background (watts/meter^2)
E_T	-	The Average Irradiance of Target (watts/meter^2)
F_n	-	Number of Frames to Resolve Target with Resolution R_o
F_p	-	Number of Frame Samples During t_d
F_R	-	Sensor Frame Rate (Hertz)
$H_D(f_x,f_y)$	-	Transfer Function of a Square Detector
$H_O(f_x,f_y)$	-	Transfer Function of Optics
$I_e(u,v)$	-	Radiant Intensity From Point (u,v) (watts/steradian)
j	-	Imaginary Number $(j)^2 = -1$
L	-	Footprint(L^2 = Area Subtended By DFOV) (meters)
LET	-	Largest Expected Target, Determined From Aircraft Wingspan (meters)
M	-	Blackbody Emittance (watts/meter^2)
N	-	Width of Focal Plane in Detectors
P_{DFOV}	-	Power in the DFOV (watts)
$\overline{P}_{DFOV}$	-	Spectrum of P_{DFOV}

$P(u,v)$	-	Power Incident on Sensor from Point (u,v) in Object Space (watts)
R	-	Range or Altitude (meters)
R_i	-	Resolution of Data Extracted from an In-Phase Target (meters)
R_o	-	Resolution of Data Extracted from an Out-of-Phase Target (meters)
$REM()$	-	The Fractional Portion of the Enclosed Expression
$S_{i,j}$	-	Signal From Detector at Focal Plane Coordinates (i,j)
SFOV	-	Sensor Field-of-View
T_a	-	Transmission of the Atmoshpere
t_d	-	Target Dwell Time (seconds)
T_e	-	Temperature of Aircraft Engine (Kelvin)
T_o	-	Temperature of Air Flow (Kelvin)
T_s	-	Temperature of Aircraft Skin (Kelvin)
t_x	-	Time Difference Between Target Samples (X-Windows)
t_y	-	Time Difference Between Target Samples (Y-Windows)
V_T	-	Target Velocity (meters/second)
V_{Tx}	-	Target Velocity, X-Component
V_{Ty}	-	Target Velocity, Y-Component
W_e	-	Window Length Error
$X_i(k)$	-	X Vectors generated by Summing Frames (K) of Focal Plane Data in the i Direction
$\overline{X}()$	-	X Feature Vector
$Y_j(k)$	-	Y Vectors generated by Summing Frames of Focal Plane Data in the j Direction
$\overline{Y}()$	-	Y Feature Vector

Abstract

A unique method to extract and classify aircraft targets from low-resolution staring infrared data is presented which uses sequences of sensor frames to obtain target signature samples. The classifier is based on a target feature description consisting of X and Y energy projections of the target. The target information is first extracted from sensor data through sampling windows consisting of contiguous rows and columns of detectors in the path of the expected target. The samples are then reduced to remove degradations from noise and sensor optics. The classifier assumes that approximate target position and velocity are available from a target detection and track subsystem. Tests were performed using a software sensor simulation that includes optical blur and sensor motion models. The classifier is shown through simulation to successfully extract and classify various transport and fighter class aircraft targets with sensor footprint resolution of 11 to 45 meters.

the in-phase case and given as

$$R_i = \frac{V_T}{F_R} = \frac{L}{F_p} \tag{11}$$

where:

R_i = the resolution of the in-phase target samples in meters

The resolution of the target in the previous example crossing a 9 meter footprint would then be 9/4 or 2.25 meters. However, in the more likely case, if the target is "out of phase" with the sensor, further resolution is possible. The amount of increased resolution is fixed for a rational F_p and is given as

$$R_o = R_i * \mathrm{REM}\left(\frac{L}{R_i}\right) \tag{12}$$

where:

R_o = the resolution of the out-of-phase target samples in meters

REM() = the fractional part of the enclosed expression

and the number of frames F_n required to resolve the target is given as:

$$F_n = \frac{1}{\mathrm{REM}(L/R_i)} \tag{13}$$

For example, consider a one-dimensional case with a sensor having a footprint of 20 meters, and a frame rate of 40 Hz in which a target with a velocity of 250 meters/sec passes through a single DFOV. The sample width is then 250/40 or

to the detector boundary:

$$t_d = \frac{L}{V_t} \tag{9}$$

where:

t_d = target dwell time
V_t = is target velocity

When the target dwell time is related to the system frame rate, an important parameter which describes the number of times a point on the target will be sampled as it moves across the detector footprint is generated. This parameter is designated F_p and is given by,

$$F_p = F_r \, t_d \tag{10}$$

where:

F_p = the number of frames during the dwell time of the target
F_r = the system frame rate (Hz)

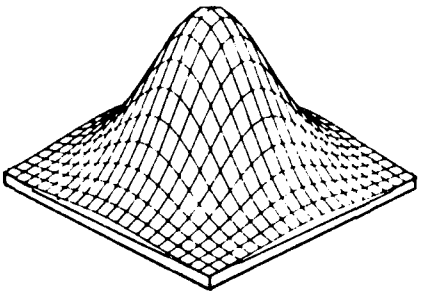
Thus, a target which has a dwell time of 0.1 seconds passing through an DFOV with a frame rate of 40 Hz can expect to have 40*0.1 or 4 frame samples per detector.

The ultimate resolution of the acquired data is not fixed by the sample rate, but by the relative phasing of the target velocity vector to to the DFOV. For the previous example, the target may be considered "in phase" when the target crosses the footprint in a integer number of samples (F_p is an integer). Thus, the resolution is then fixed for

outlined in later chapters.

.018	.082	.135	.082	.018
.082	.368	.607	.368	.082
.135	.607	1.00	.607	.135
.082	.368	.607	.368	.082
018	.368	.135	.082	.018

Digital Model



Continuous Model

Figure 2-3. Example of Simulated Optical Blur Function.

The combined effect of equations (6) and (7) on equation (2) gives the following expression describing the spatial transformations to the incident energy on the detector,

$$S_{i,j} = \iint \bar{P}_{DFOV} H_d(f_x,f_y) H_o(f_x,f_y) \exp[2j(f_x x + f_y y)] df_x df_y \quad (8)$$

where:

$S_{i,j}$ = the output of detector at coordinates (i,j)
P_{DFOV} = the spectrum of P_{DFOV}

The temporal target response of a mosaic sensor depends on the target speed and the crossing geometry. The time in which the target takes to cross the footprint of a single detector is referred to as the target dwell time; the dwell

Fourier transform is given by (Ref 4),

$$G(f_x,f_y) = \iint g(x,y)\ \exp[-j2\pi\ (f_x x+f_y y)\ dxdy \qquad (4)$$

and the corresponding inverse,

$$g(x,y) = \iint G(f_x,f_y)\ \exp[\ j2\pi\ (f_x x+f_y y)\ df_x df_y \qquad (5)$$

The detector is modeled as a square aperture (of area = L^2) with uniform response whose transfer function is given by,

$$H_d(f_x,f_y) = \frac{\sin(\ Lf_x\ )}{Lf_x}\ \frac{\sin(\ Lf_y\ )}{Lf_y} \qquad (6)$$

and the optical transfer function which is approximated by a two-dimensional gaussian is given by,

$$H_o(f_x,f_y) = \exp[-2\ f_x^2\ \sigma^2]\exp[-2\ f_y^2\ \sigma^2] \qquad (7)$$

where:

σ^2 = the approximate width of a detector for a matched optic configuration

Figure 2-3 shows the gaussian point-spread function (PSF) used in the computer simulation along with the relative shape of the point-spread function compared to the detector. This function is convolved with input scenes to simulate the blurring of diffraction-limited optics. In effect, the PSF acts as a low-pass filter attenuating high spatial frequencies present in input scenes. Removal of the PSF will play a significant part in the recovery of the target signal

of data, with the time between samples referred to as the frame time. The mapping of energy from the scene onto the focal plane can be model in terms of the scene radiant intensity I_e. The scene is modeled in 2-D coordinates (u,v), where the energy transfer to the sensor from point (u,v) is

$$P(u,v) = I_e(u,v)T_a\Omega \tag{2}$$

where:

$P(u,v)$ = power in watts at the sensor from point (u,v)
$I_e(u,v)$ = radiant intensity in watts/steradian from a point (u,v) in object space
Ω = represents the solid angle where $\Omega = A_o/R^2$
T_a = is the transmittance of the atmosphere which will not be modeled in subsequent calculations

It is convenient to express the two-dimensional nature of the energy transfer in terms of a number of point intensities since the simulation will model each point in the image as a integration of a infinite number of points in a scene.

$$P_{DFOV} = \iint_{DFOV} P(u,v)\ dxdy \tag{3}$$

The transfer functions of both the optics and detector are assumed to be linear and shift invariant such that the detector signal can be modeled in the Fourier domain as a multiplication of the 2-D Fourier transform of the signal with the transfer function of the optics and the transfer function of the detector aperture. The two-dimensional

Energy radiated from the target and the background is imaged onto a mosaic array of detectors referred to as a focal plane. Each detector receives energy from an area subtended by the detector field-of-view (DFOV) as shown in Figure 2-2. The spatial energy distribution on the surface of the detector can be considered a convolution with the transfer functions of the sensor system and the incident target and background energy. Similar models have been described in (Ref 1,15,18), however the target under investigation is usually represented as a point source or a gaussian distribution. Since the resolution of the sensors supported by this effort range from 10 to 100 meters, it is important to model the target as a two-dimensional signature having specific shape characteristics which will be described later.

Several sensor specifications have a direct influence on target acquisition. These include detector geometry, optics, range, and noise. The extent of the DFOV is usually referred to by the area in the scene subtended known as the detector "footprint". Thus, the footprint is represented by

$$L = R \sin(\theta_{DFOV}) \tag{1}$$

where:

L = width of footprint in meters (area = L^2)
R = is the range (altitude)

During observation, the focal plane is sampled at a specified rate to generate data. Each sample is referred to as a frame

The system effects can be further separated into sensor geometry and motion, optical and detector aperture functions and transmission, and background noise both direct and induced by sensor instabilities. Models describing sensor performance and associated noise characteristics have been a topic of research for several years (Ref 1,7,8,12,16,18). The intent of this section is to review some of these models as they apply to this specific problem, and to support simulation of these models in later sections.

Sensor Scenario

The scenario under study consists of a high-altitude down-looking staring mosaic sensor and a target in cruise flight passing through the sensor field-of-view (SFOV).

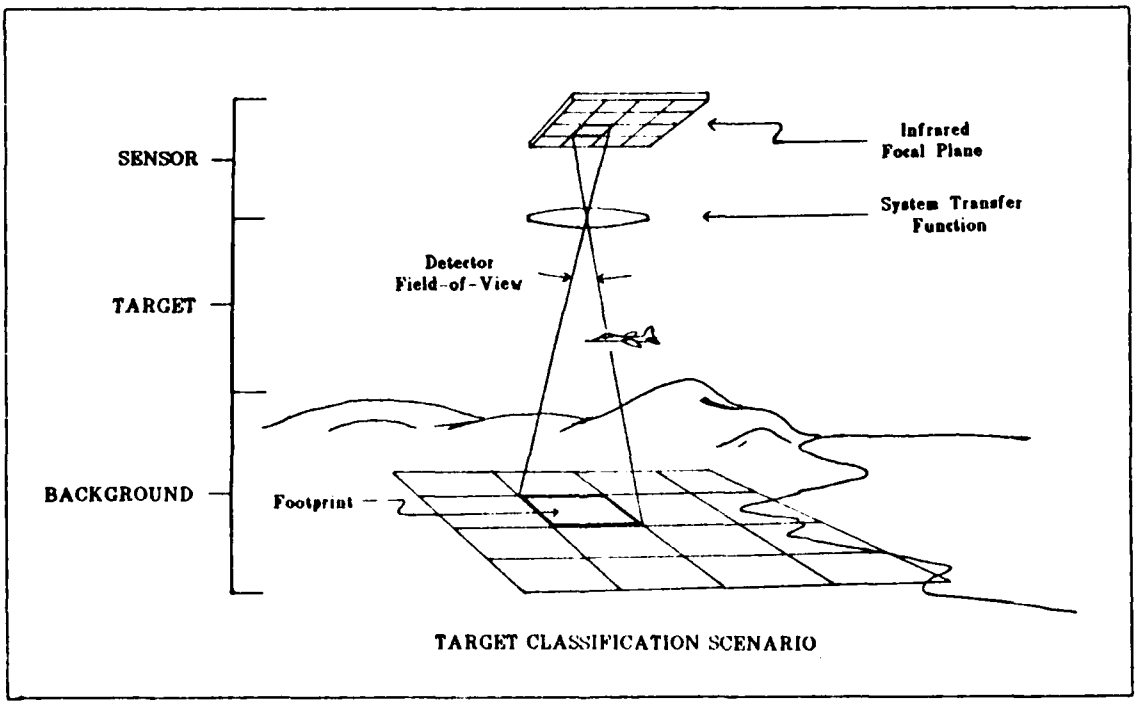


Figure 2-2. Staring Sensor Scenario.

II. GENERAL THEORY

Introduction

To properly analyze the signal generated from a staring mosaic sensor, it is necessary to model the essential characteristics of the sensor, background, and target. This analysis of the interaction between the target signal and various degradation effects inherent in the acquisition process provide a basis for developing an approach to counteract these effects. Use of a staring infrared sensor for target classification presents a number of problems which impact the quality of the acquired signal. The signal ultimately used for target correlation must pass through several stages of acquisition each of which contribute various levels of degradation to the original signal. These noise contributions are generally separated into those caused by the scene which the sensor is viewing, and the sensor itself. An overview of this process is shown below in Figure 2-1.

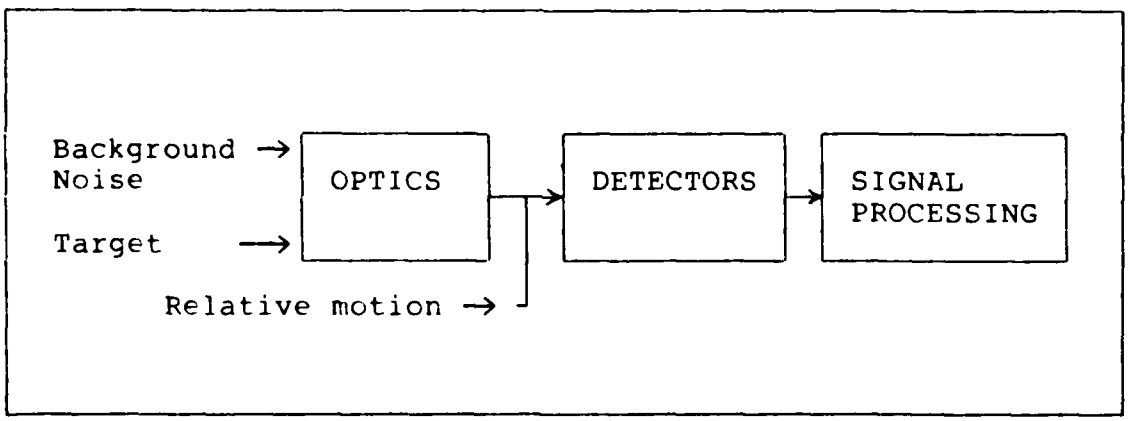


Figure2-1. Staring Sensor Data Acquisition.

derived and implemented to obtain a performance characterization relative to a limited number of target types. Finally, a simulation modeling critical aspects of staring sensor signal acquisition process is developed to generate data to test the aforementioned target extraction scheme.

Presentation of this research will proceed with a discussion of general theory concerning the scenario dependent sensor, target, and background considerations. This is followed by a discussion of a target feature set derived from image sequences and later applied to a feature extraction algorithm. This algorithm is then implemented and tested in a computer simulation to determine the feasibility of low-resolution target classification. Lastly, the results and analysis of algorithm performance are presented with recommendations for continuing study.

Materials and Equipment

All thesis research was performed in the Signal Processing Laboratory at AFIT utilizing the Data General Eclipse S/250 and Nova computers and the Octek Image Analyzer which can acquire and display a 320x240 monochrome image display with 4 bit resolution. Target and background data consisting of photographs and models are input to the Octek digitizer by a video camera. Programming was accomplished using FORTRAN IV and FORTRAN V when possible. Existing support software is utilized as available.

However, it is expected that a certain amount of training is required to optimize any feature set correlation scheme and that a practical implementation of these techniques would utilize actual target signatures.

Specification of sensor noise contributions will be important in evaluating the classification potential of target acquisition techniques. The sensor itself contributes a certain amount of noise to the image in the form of an optical transfer and detector aperture function (blurring), jitter and drift (movement of the sensor), and detector variations. It is assumed that contributions due the detector electronics will be background limited and therefore neglected. Likewise, temporal variations due to scintillation and other sources will not be included. However, contributions due to spatial gradients in the background scene are included to a limited extent.

Approach

To understand the impact of various degradation mechanisms inherent in the staring sensor, the target, background, and sensor models are represented initially in a simplified manner to facilitate identification of a useful target feature set. The extraction of a feature set from the limited amount of information generated by a target is important to the success of the classification scheme and is studied extensively. A feature extraction method is then

Algorithms for target detection are not included due to thesis time constraints. Therefore, any implementation of these techniques will require hand-off information from a target detection scheme concerning target position and relative direction.

Assumptions

Since the distance between the sensor and the target is assumed to be relatively large (on the order of 50-100 km), the target may be represented by a two-dimensional projection. An aircraft for example, would appear as a flat 2-D form as observed from space. Therefore, all target classification techniques considered in this study will utilize a 2-D projection of the target on an earth background. This eliminates the need for 3-D geometric transformations of objects thereby reducing the recognition problem to a single aspect angle of the target. Given this simplification, the major considerations for a recognition methodology will be target orientation and size invariance.

It is expected that the target signature will be a known quantity such that features such as size and shape may be utilized to identify the target. Target signatures used in this study are intended to represent a first-order approximation to actual signatures which are unavailable due to security classification. It is acknowledged that this representation could affect the performance analysis.

Scope

This effort is primarily concerned with the feasibility of extracting useful target information from low-resolution image sequences and the investigation of various sensor and scenario dependent parameters which impact target classification. Since the purpose of this research is to support the Foreign Technology Division's (FTD) need to analyze specific sensor data, scenarios used in this study consist of a high altitude IR staring sensor with a nadir view of an earth background. Other geometries are not considered. Scenes are generated by computer simulation which model critical effects of background clutter, target motion and sensor optics. Background scenes consist of visible-band photographs which are intended to provide a first-order approximation of an infrared cluttered background. The primary objectives of this effort are:

1. Demonstrate feasibility of classifying various aircraft types from low-resolution data.

2. Investigate various scenario dependent parameters which impact the target recognition process.

3. Perform a performance analysis to ascertain the potential application of techniques developed herein to a practical system application.

4 Develop generalized algorithms which may be applied to specific sensor data as required by FTD.

Thus, the target appears as a single dot or small blob and is usually characterized and identified by its amplitude, direction, and sometimes velocity. However, target motion across the sensor field-of-view can provide enough additional information to allow target classification. Although using image sequences to enhance resolution is not new, this application provides a unique approach to the target classification problem and may serve to reduce false alarm rates plaguing current system performance.

Problem

The problem to be addressed by this effort is the development of a methodology to classify low-resolution targets based on features extracted from temporal variations in sequences of infrared imagery. This problem can be divided into four parts:

1. Detection and location of potential targets in a noisy scene.

2. Extraction of target information by observing target translation through sequential frames of data.

3. Transformation of acquired data into a useful set of features unique to the target.

4. Correlation of derived features with those of known targets.

LOW RESOLUTION TARGET CLASSIFICATION FROM A STARING INFRARED SENSOR

I. Introduction

Background

A high-altitude or space-based infrared sensor using a staring mode of operation offers increased resolution and sensitivity over current scanning systems for surveillance missions. A staring sensor employs millions of detectors compared to several thousand for the scanning system and can look or "stare" at one point for relatively long periods of time, thus, collecting more incident energy during observation. The technology for developing the detectors and associated processing for staring systems is rapidly maturing and prototype systems are scheduled for test over the next several years. The Foreign Technology Division (FTD) at Wright-Patterson AFB has a requirement to analyze data provided by these prototype staring systems and to develop the associated processing techniques which will automatically locate and identify low-resolution targets of interest.

Although many processing techniques have been presented in the literature which can detect a target in noisy backgrounds, these techniques consider targets of interest to be "point" targets. Therefore, information pertaining to the target form or type is largely ignored.

6.25 meters and 20/6.25 or approximately 3 frames would be required to sample the data. However, if more samples of data were used, the resolution could be increased to R_o = 6.25 REM(20/6.25) or 1.25 meters with 5 frames of data (illustrated in Figure 2-4.) This is accomplished by summing the samples from subsequent detectors with that of the first detector. The result is an equally spaced sampled energy profile of much higher resolution then samples from a single detector. Theoretically, for an irrational F_p, the potential resolution would be infinite for an infinite number of frames, i.e. $\lim_{F_N \to \infty} R_o = 0$. Practically of course this would be impossible to implement. Therefore, the irrational F_p is aproximated by a rational value with little error induced due to the low frequency energy curve which the data approximates. The application of this resolution enhancement is a key factor in determining a useful feature extraction method as will be shown in the next chapter. The practical application is limited however due to sensor noise and other signal degradation phenomena depending on the number of detector samples to be reduced to a single sample.

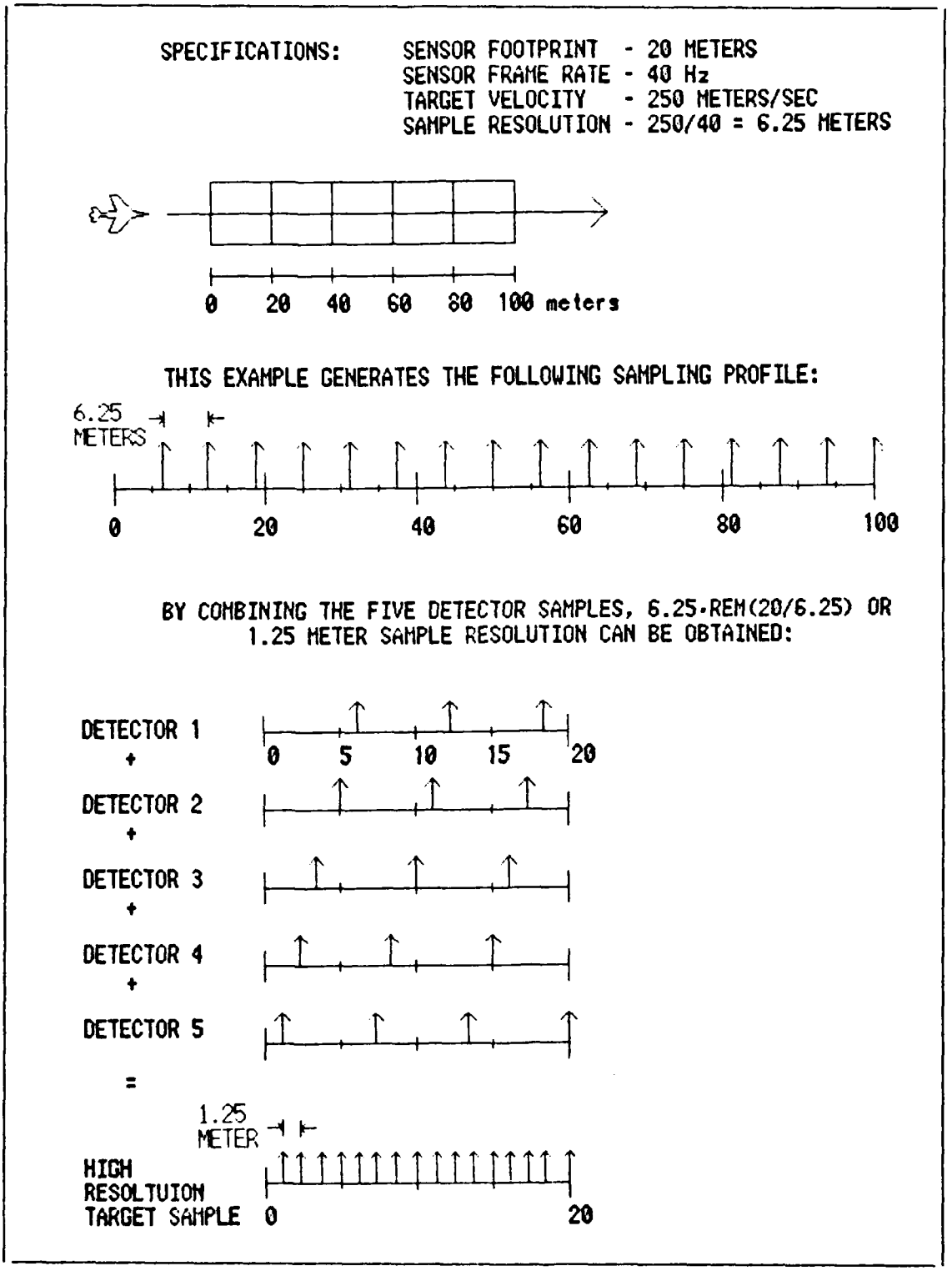


Figure 2-4. Example of Out-Of-Phase Target Sampling

Staring Sensor Noise Sources

The two-dimensional scene observed from the staring mosaic space platform can contain many features that tend to obscure or degrade the sensor's target detection capability. When the target is weak relative to the background, (such as with an aircraft) the majority of the incoming signal is considered to be noise (Ref 15:103). Although the potential to detect weak targets against structured background is a primary advantage for staring IR sensors, the background can not be eliminated completely due to sensor drift, jitter, clutter, cloud movement and electronic noise.

Small instabilities in the sensor can lead to background spatial variations being interpreted as temporal change resulting in an AC signal from the detector. Therefore, the earth background is considered a noise source. Generally, signal processing must be performed to raise the level of the signal relative to the level of the noise. The type of noise or background suppression technique used is dependent upon the nature of the sensor noise sources.

There are four major sources of noise in a staring mosaic sensor. These include photon fluctuations of incoming radiation from both targets and background, temporal variations in background radiance, spatial variations due to movement between the sensor field-of-view (SFOV) and the background, and finally, noise from the detector and associated electronics. These noise sources are discussed in

detail in Pohlman (1976) and Pollock (1981). Typically, sensors are designed to minimize noise sources while enhancing the target signal. However, it is extremely difficult to design against all background situations.

A prominent noise source which every detection system must deal with is clutter. According to Pollock (1980), "Clutter may be defined as any signal which must be processed to some degree to determine if it is a target or a non-target". Sources of clutter depend upon the sensor and viewing conditions, but may range from clouds and glint from ocean specular reflection, to structured patterns of corn fields and industrial heat sources such as smoke stacks. Schemes for suppressing background clutter include adaptive thresholding based upon the statistical nature of the background and temporal filtering to subtract slow moving backgrounds from fast moving targets. A technique for suppressing clutter due to motion of the sensor (jitter) is presented in Mazaika (1982) in which he develops a model for clutter leakage due to gradients in the background radiance. Jitter is a significant effect in many sensor designs and may severely limit the detection capability of the sensor. Measurement of the influence of temporal and spatial variations is usually accomplished by spectral analysis of the background signal.

Another important effect of the background is its relative temperature contrast with the target. Target

information integrated over the detector footprint can be lost due to the masking of background radiance with that of the target signature. This process is not reversible. To see this, consider the 2-D signal imaged onto the focal plane without sensor transfer effects. The signal on the detector at time t is given by,

$$S_{ij}(t) = A_T(t)E_T + [L^2 - A_M(t)]E_{BKG}(i,j) \qquad (14)$$

where

$S_{ij}(t)$ = the signal on detector (i,j) of the focal plane
$A_T(t)$ = the area of the target in the DFOV at time t
$A_M(t)$ = the area of the radiant background masked by the target (varies from 0 to $A_T(t)$) at t
E_T = the average irradiance of the target in watts/meter^2
E_{BKG} = the average irradiance from the background in watts/meter^2 on detector (i,j)

The effect is illustrated in Figure 2-4. Given a 3 by 3 array of detectors, a single cloud partially obscures the field of view of the central detector (all other background is negligible). For illustration, the cloud has a uniform signal of level 5 watts/meter. Using Equation (14), the signal due to the background cloud is $5L^2$. If the target (also with a uniform signature of 5 watts/meter) passes through the center detector along path 1, the target signal will appear on the detector rising to a maximum amplitude of $5A_T+5L^2$. However, if the target takes path 2, the relative contrast is zero since $5A_T+51^2-5A_T = 5L^2$. Therefore, no target information will be present on the center detector

using path 2. This clutter severely degrades effect to the target signature in low signal to noise ratio scenarios.

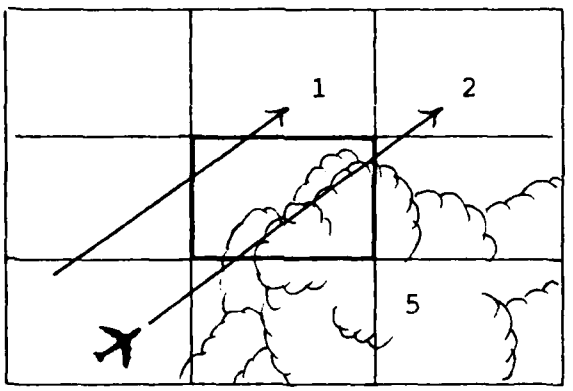


Figure 2-5. Signature Degradation Due to Background Contrast

Target Model

The model for the target signature should be representative of expected target signatures in the 8-12 micron band. The target signature for this region is largely due to the target thermal temperature and resulting blackbody emittance. From a high-altitude sensor however, the actual signature depends on the radiance contrast between the target and its immediate background. Therefore, the spatial extent of the target can be represented as a 2-D silhouette whose shape is invariant. Similar approaches to representing thermal targets are presented in several studies analyzing stationary tank and ship targets (Ref 2,5,6,11,19). For this analysis, the target motion is restricted to 2-D translation and rotation in a plane parallel to the 2-D image plane. For

the purposes of classification, it is assumed that the target signature is a known quantity. No attempt has been made to study variations in the target signature and it is assumed that models are available (or can be developed) for a practical implementation. For a first order approximation, the target signature results from the contrast between the target and background. This can include both positive and negative temperature contrasts which can vary as the target moves across a highly structured background. The target is modeled here as a relatively uniform intensity distribution over the target shape tapering at the edges. A more detailed model would account for the the difference in thermal signature due to the engine temperature and the adiabatic heating of the aircraft skin. For this model, the engine is represented as a graybody whose radiance is proportional to a blackbody at engine temperature T_e. The temperature of the aircraft skin T_s is given by:

$$T_s = T_o[1 + r\frac{(\gamma - 1)}{2} M^2] \tag{17}$$

where:

T_s: Adiabatic wall temperature in Kelvin
T_o: Temperature of Flow
r : Recovery factor (.85 - 1)
γ : Specific heat ratio
M : Mach number

Both temperatures are now used to calculate the emmittance

from a black body using Plancks law Ref(9:35),

$$M = \frac{2 * hc^2}{\lambda^5} [\exp(hc/\lambda kT) - 1]^{-1} \text{ watts/m}^2 \qquad (15)$$

where

M = the blackbody emmittance in watts/meter
h = Plank's constant (6.6256×10^{-34} watts/sec^2)
c = speed of light (2.997925×10^8 meters/sec)
λ = wavelength (meters)
k = Boltzmann's constant (1.38054×10^{-23} watt sec K^{-1})
T = Temperature in degrees Kelvin (either T_e or T_s)

the intensity of an aircraft engine and skin can be modeled as Lambertian,

$$I_e = \frac{\epsilon M A_e}{\pi} \text{ watts/steradian} \qquad (16)$$

Thus, the actual signature of a target may be inferred from the specific features of the target such as the number of engines, placement, and the aircraft structure area. A similar method to represent targets is presented in (Ref 2).

The aircraft signatures used in this study were modeled with very little detail. Although this can be attributed to the resolution of the hardware used in the simulation, it is assumed to be representitive of a typical scenario. This can be asserted for two reasons:

1. At long ranges, details represented by small temperature differences are unresolvable.

2. The spatial resolution of the sensor array tends to eliminate high spatial frequencies of the target.

The targets used in the simulation are generated by taking pictures of plastic scale models of aircraft using a video camera. The models, which are painted flat black, are placed on a white background and rotated to the desired orientation before acquiring the picture. After digitization, the graylevel is adjusted as necessary to match the desired signature characteristics. The following pages show five of the target models used in the simulation. Each target is shown in its original digitized form, and in its reduced form typical of the targets used in the data base of the simulation.

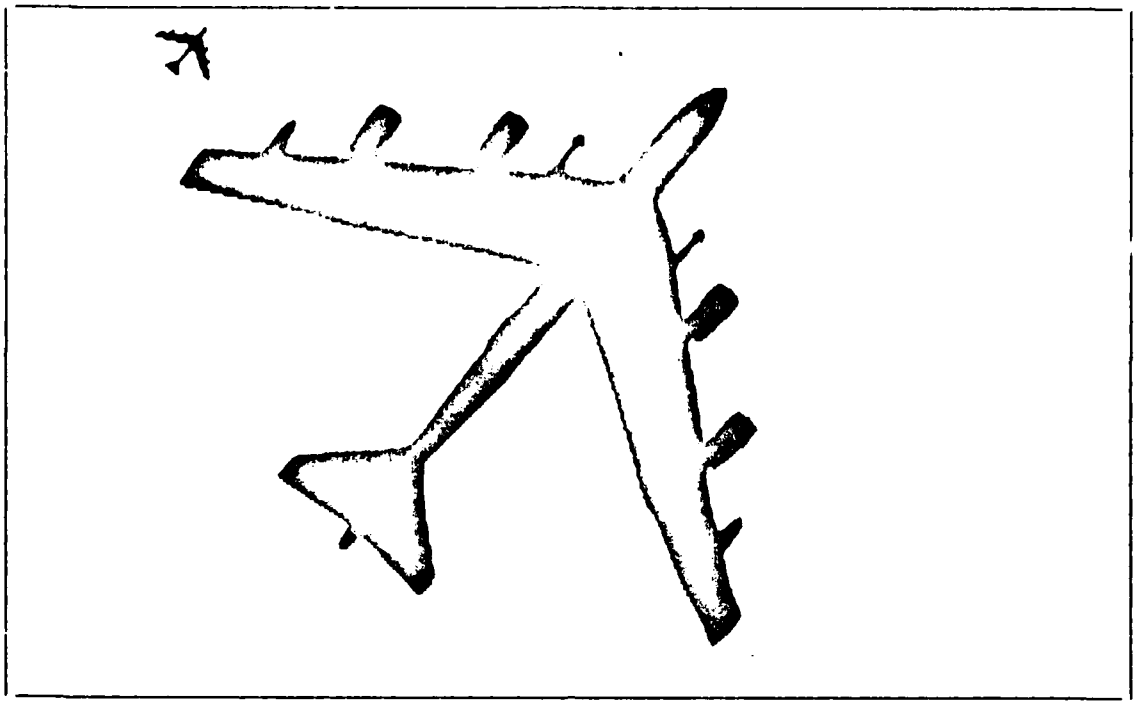


Figure 2-6. B-52 Target Model

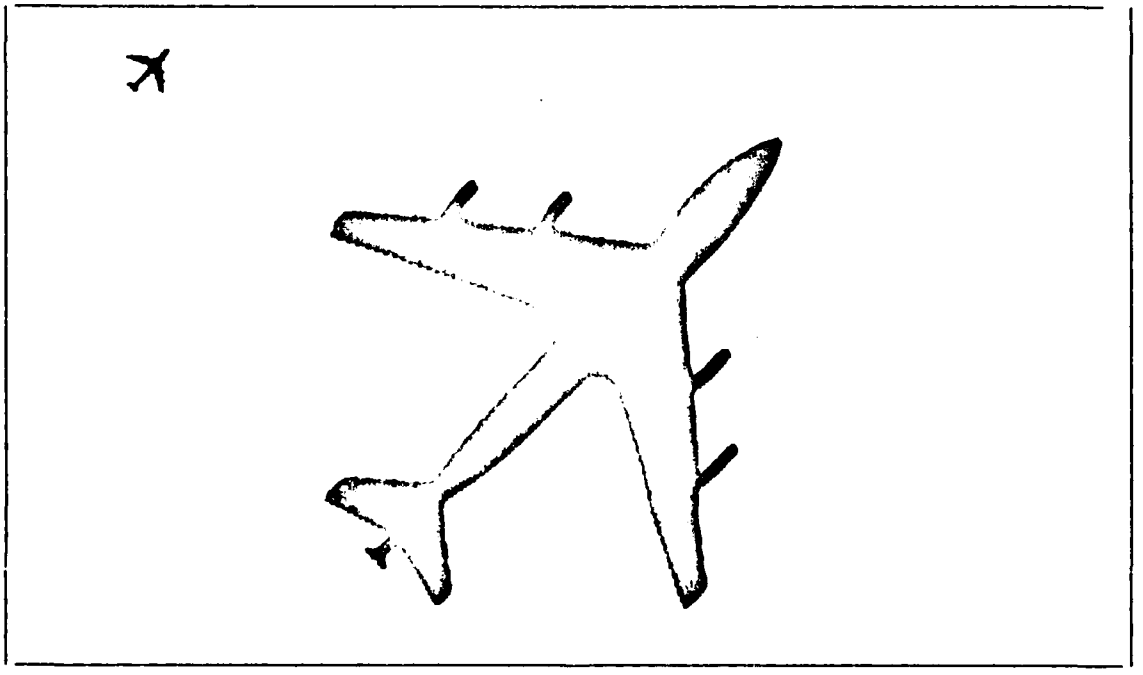


Figure 2-7. KC-135 Target Model

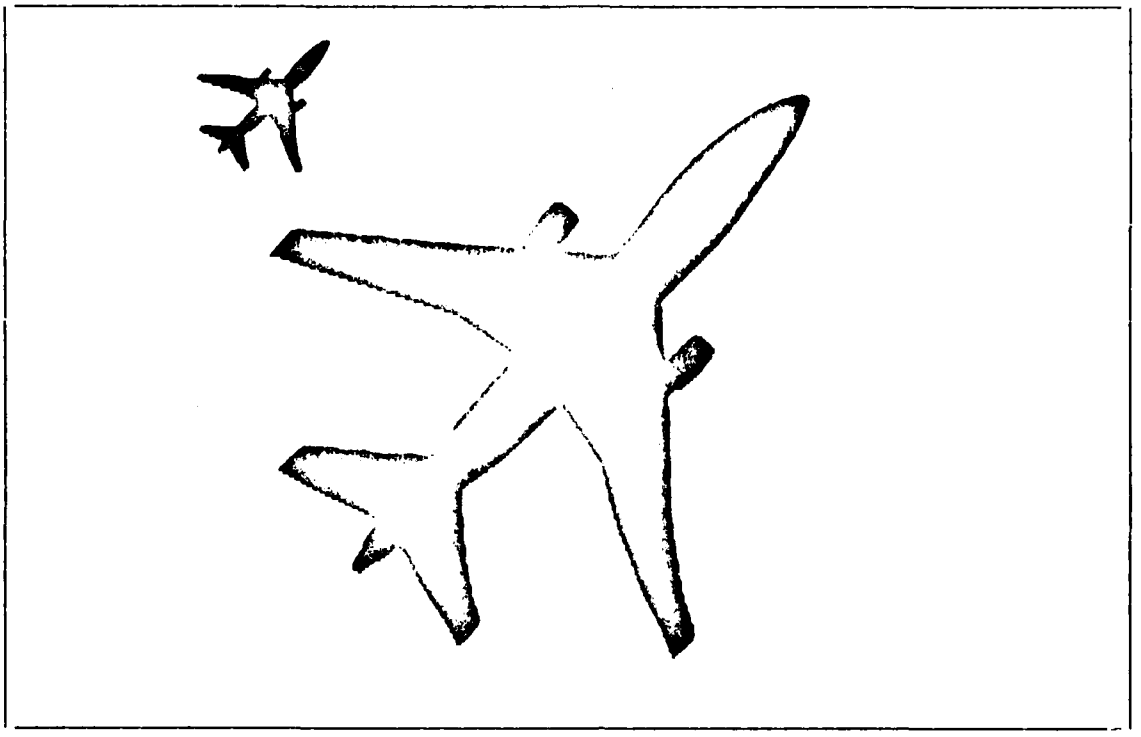


Figure 2-8. DC-10 Target Model.

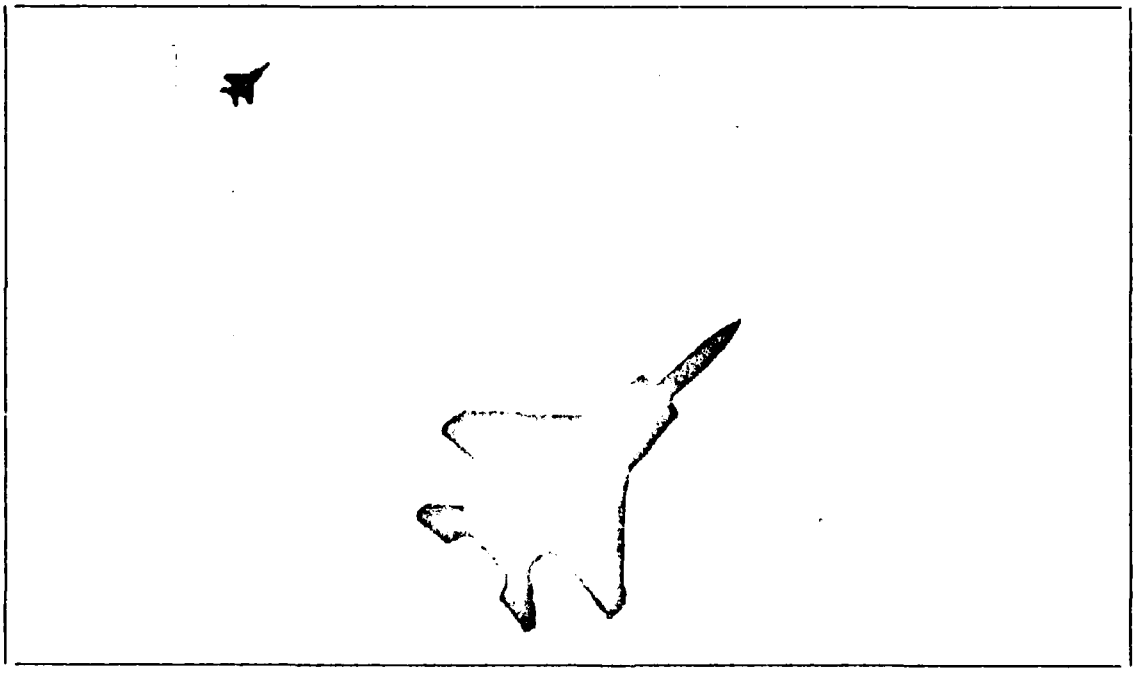


Figure 2-9. F-15 Target Model.

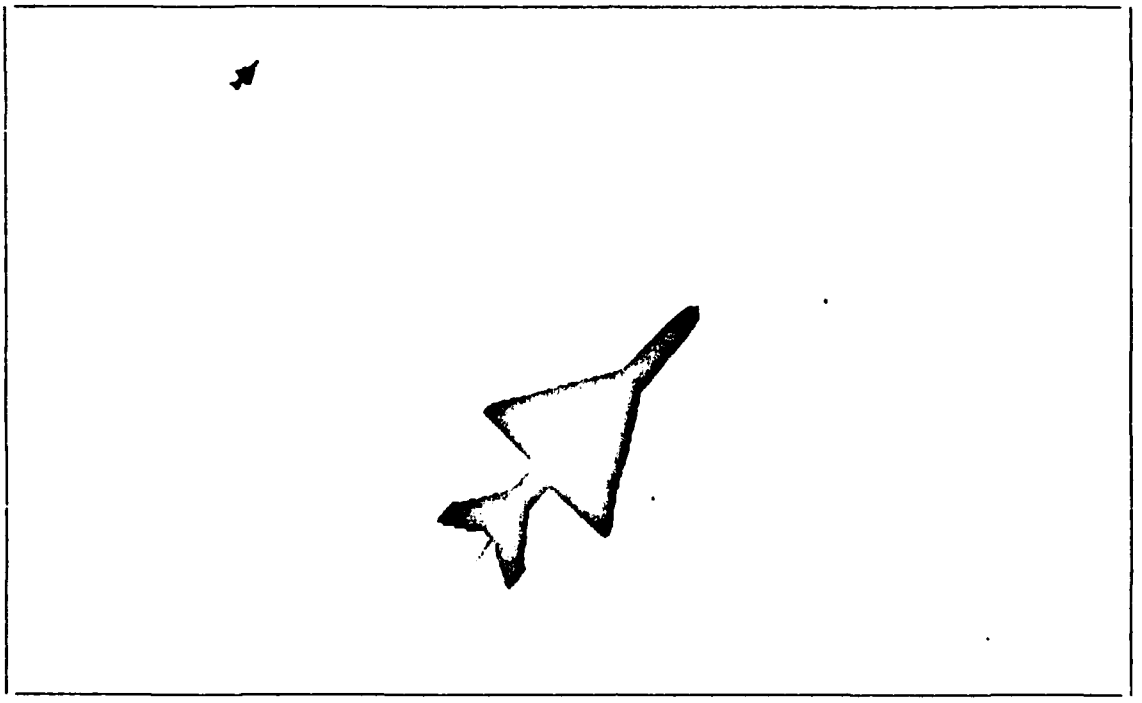


Figure 2-10. MIG-21 Target Model.

III. FEATURE SELECTION

Introduction

There are many ways to represent the shape and intensity a target. As expected, the features chosen to represent a target have a direct influence on the potential success of the classification scheme. The ultimate goal of any feature selection process is to identify those features which maximize separability of the expected target classes. Other considerations include invariance to noise, ease of implementation, and computational cost. This chapter will focus on deriving a feature set which will use to the maximum extent, the target shape information extracted from low-resolution imagery.

Background

Current methods to represent shape and intensity information include Fourier boundary descriptors and invariant moments as described by several studies (Ref 2,5,19). These methods use features such as height-to-width ratio, perimeter-to-area, normalized intensity moments, and normalized silhouette moments. However, the targets used in the referenced studies were of relatively higher resolution then those considered here. An accurate description of a perceived target from a staring sensor can be at best, a "shaped blob." Thus, the scenes considered here have a

maximum of 10 pixels on the target, with a more likely resolution of 3-4 pixels. Even prominent target features such as wings and tail are difficult to visualize at this resolution. This is illustrated below in Figure 10, which shows five frames of a B-52 at high resolution along side a comparable low-resolution signature as would be observed by

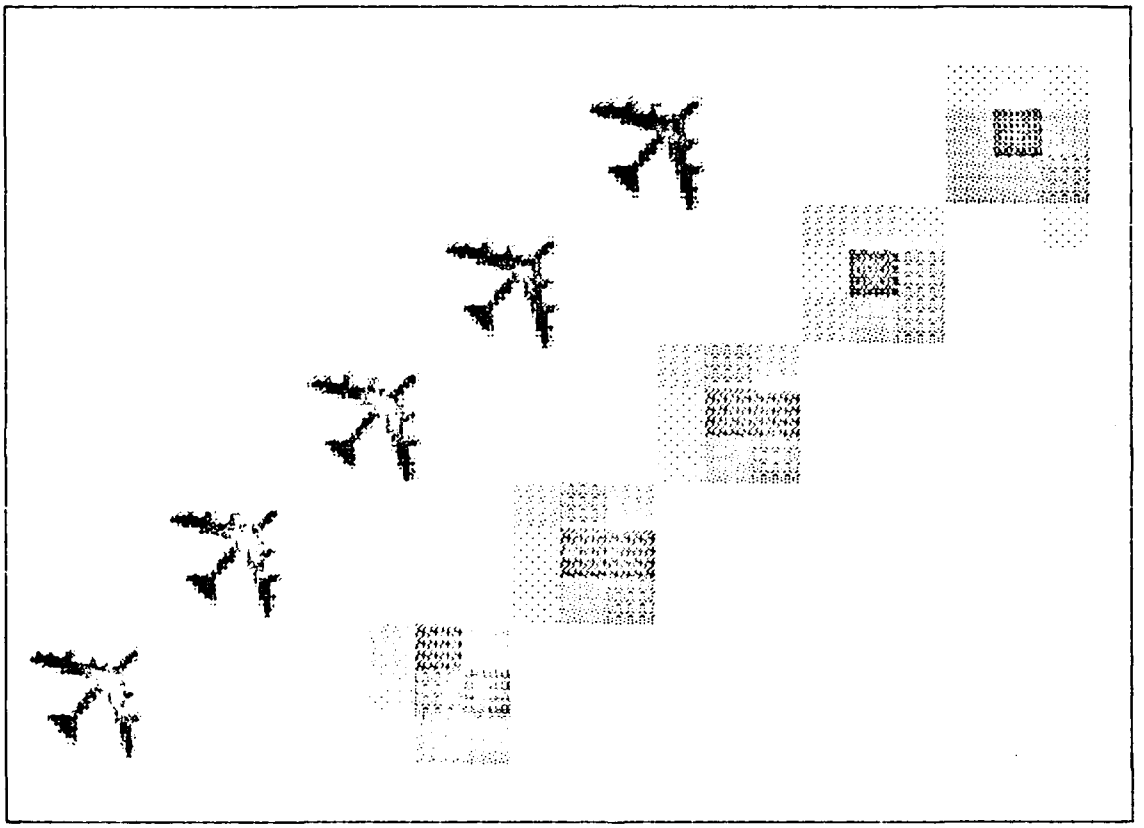


Figure 3-1. Target Signature Sequence as Obtained From a Staring Sensor

the staring sensor. The graylevel variations in the low-resolution signature are due to small changes in the relative position of the target with respect to the focal plane. While this resolution may seem high for the typical staring

Redundant Window Sampling

Aquisition of accurate target samples is obtained in preprocessing using redundant sampling windows with limited size. The number of windows required in each direction is arbitrary and depends on the spatial, temporal, and contrast of background clutter in the local area subtended by the window. For homogeneous, high-contrast scenes , two to three windows may be sufficient. For backgrounds used in this study, five to ten windows in each direction were used. An example of contrast deficiencies induced by high spatially variant scenes is shown in Figure 4-3. In the illustration, a portion of the target crossing detector 2 of detector row 2

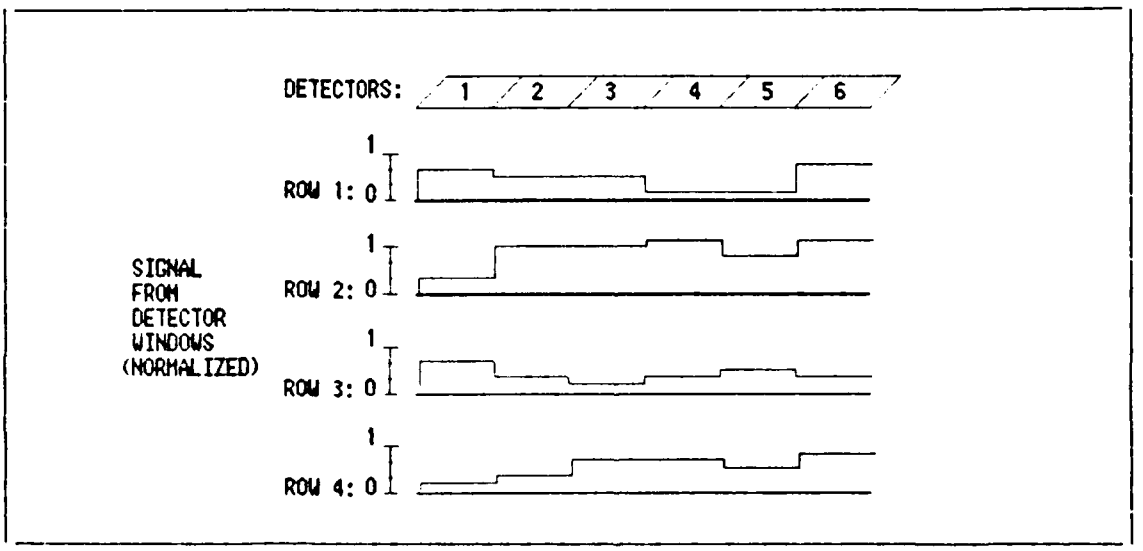


Figure 4-3. Background-Induced Signal Variations From Parallel Windows

will be largely lost due to poor relative contrast. However, much of the same portion of the target may pass through the DFOV of detector 3, window 3 where greater contrast may be

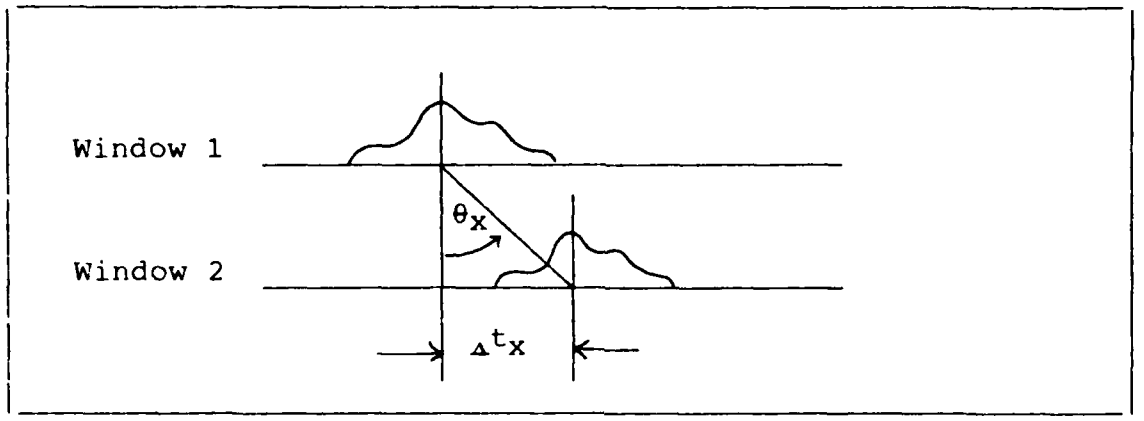


Figure 4-2. Phase Relationship of Data From Parallel Windows

length as the velocity estimate converges. As shown above, t is equal to the number of frame samples divided by the frame rate F_R. Therefore, the velocity is given as,

$$V_{Tx} = \frac{L}{\Delta t_x} = \frac{1}{\sin\theta_x}$$
$$V_{Ty} = \frac{L}{\Delta t_y} = \frac{1}{\sin\theta_y} \qquad (20)$$

and the final vector as,

$$V_T = [V_{Tx}^2 + V_{Ty}^2]1/2$$
$$\text{Dir} = \tan\left(\frac{V_{Ty}}{V_{Tx}}\right) \qquad (21)$$

In low noise cases, the profiles obtained in this step may be adequate for preprocessing. However, since background clutter generally corrupts the target vector profiles, a single sample may not be sufficient. Also, several windows could be combined to obtain an increased number of target samples. Thus, a more extensive extraction scheme is needed.

The sampling window concept differs from the mathematical development presented in Chapter 3. As opposed to solving a set of linear equations, the target signal is sampled through a logical window as a convolved sum. Therefore, the vectors extracted from focal plane data will be deconvolved to solve for the target feature vector. This method is more robust than the simultaneous equation approach which suffers from ill-conditioned matrices when noise is present.

Since hand-off information from target detection is limited to a relative position and direction vector (computed from target track), it is necessary to collect preliminary information concerning target velocity and direction to make the extraction process more efficient. Thus, preliminary windows are first set-up as shown in Figure 4-1 based on the Largest Expected Target (LET) that the algorithm is expected to process. As the target passes through the sample windows, the profile is extracted from each window generating a phase relationship between parallel windows based on the target velocity and crossing geometry. This relationship is illustrated in Figure 4-2, and allows calculation of the target velocity vector given data from each axis. In the event of severe noise, this process can be iterated to an accurate velocity-direction estimate by reducing the window

Sampling Windows

Efficient extraction of the the target vectors is accomplished through use of feature sampling windows. These logical windows consist of columns or rows of selected detectors (taking advantage of focal plane geometry) continuously sampling the target profile as it passes through the window field of view. In this way, the two-dimensional target is naturally decomposed into orthogonal vectors representing the target feature vectors in a convolved form. An illustration of this technique is shown below in Figure 4-1, where two pairs of windows have been set-up over the length of the focal plane (in this case 18 by 18 detectors) just ahead of the expected target vector.

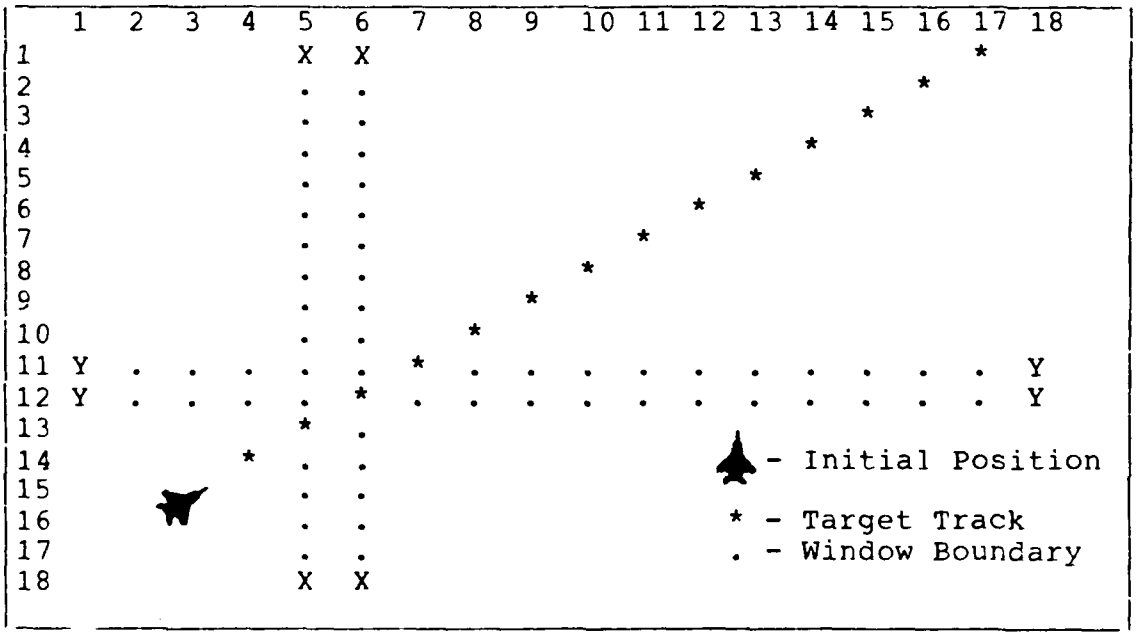


Figure 4-1. Preliminary Window Configuration.

IV. FEATURE EXTRACTION

Introduction

This chapter presents an algorithm developed to accomplish the feature extraction process. These features consist of the X-Y vectors described in chapter 3 which are later compared to a target database for classification. There are several guidelines closely adhered to in the course of this development. First is to limit the quantity and precision of the required a priori target information. It is assumed that a detection scheme exists in which an initial target position or target track record is available. Therefore, the feature extraction algorithm will assume this hand-off is information available with some limited accuracy. Since the opportunity to observe a fast moving target may be small, and also to reduce the noise effects of drift and jitter, it is desirable to acquire the target information in a minimal observation time. This observation time will vary, however, depending on target, sensor, and background signal-to-noise effects. Also, contributing to a reduction in noise effects is the sampling of a smaller portion of the focal plane during target extraction. This results in an increase in the relative signal-to-noise ratio of the target data. Lastly is the desire to accomplish the feature extraction process with a minimal amount of processing resources so that implementation of these algorithms can execute in real-time.

values. Illustrated in Figure 3-5 are two aircraft from the previous figure. A single frame of target data is shown where the signal from each detector is represented by a intensity value in the Z-axis. Below the single frames of data are the final target representations after processing of the feature vectors. The B-52 is shown with a 25 meter footprint, and F-15 with a 9 meter footprint. In each diagram, the aircraft nose and wing shape can be seen. Although this may first appear as a crude target image, the information contained in these generalized signature representations is adequate to distinguish between various target classes over a range of scenarios.

range since targets such as aircraft are symmetrical about one axis. Also, the memory required to store the vectors (approximately 100 words per vector at 2 bytes per point) is small so that a target could be fully represented in less than 10k words of memory.

A three-dimensional generalized representation of the feature vectors can be generated by multiplying the orthogonal vectors to form a 2-D spatial array of intensity

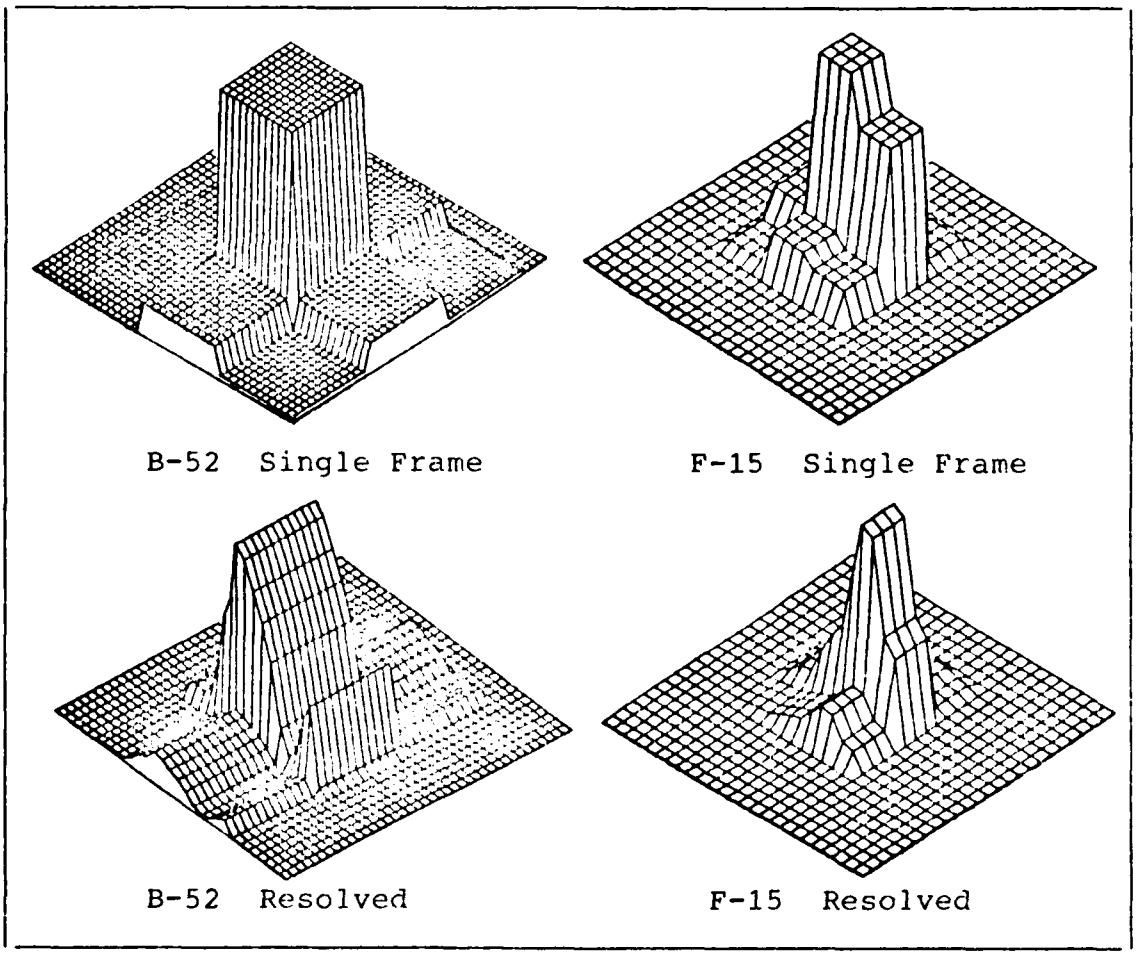


Figure 3-5. 3-D Representation of Target Feature Set.

Therefore, the target degree characteristics represented by the feature vectors are directly proportional to the resolution of the vector. This also implies that the resolution of the target vector (or the number of points in the feature set,) affects the separability of the target classes. Shown below are X-axis vectors of three targets a B-52, DC-10, and a F-15. Each target is oriented at 45 degrees and is sampled at a one meter resolution.

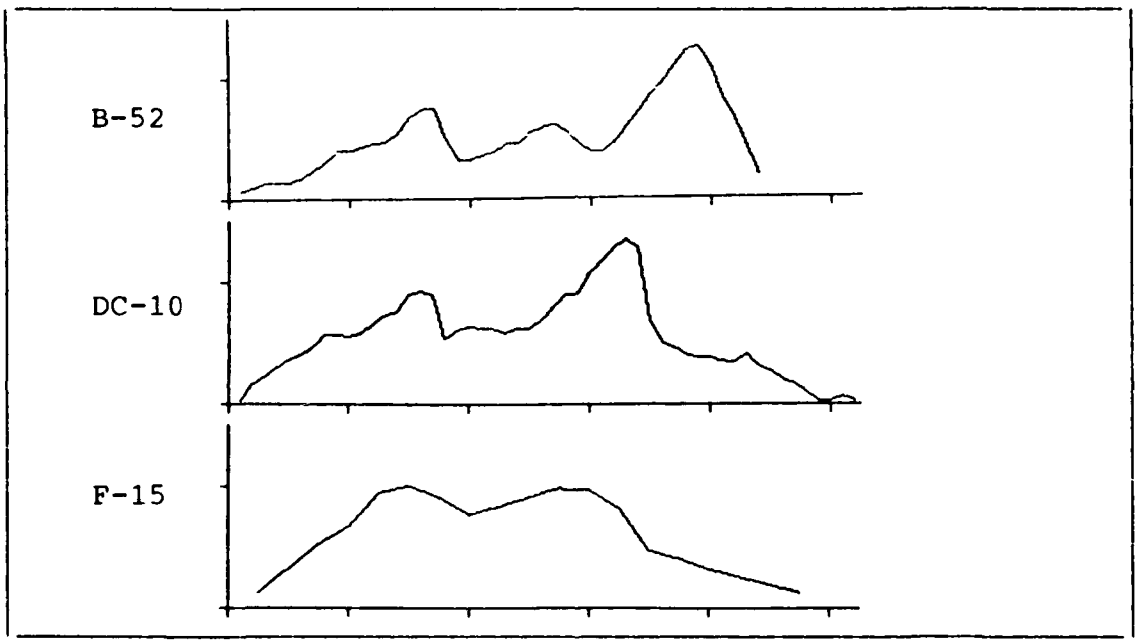


Figure 3-4. Plots of Target Feature Vectors for Three Targets.

An obvious disadvantage of this feature set is the variance of target perspective to the orthogonal sampling axis. To fully describe a potential target, a target database must maintain profiles for each target at all possible target orientations. Fortunately, most targets can be represented by profiles calculated in a 0 to 45 degree

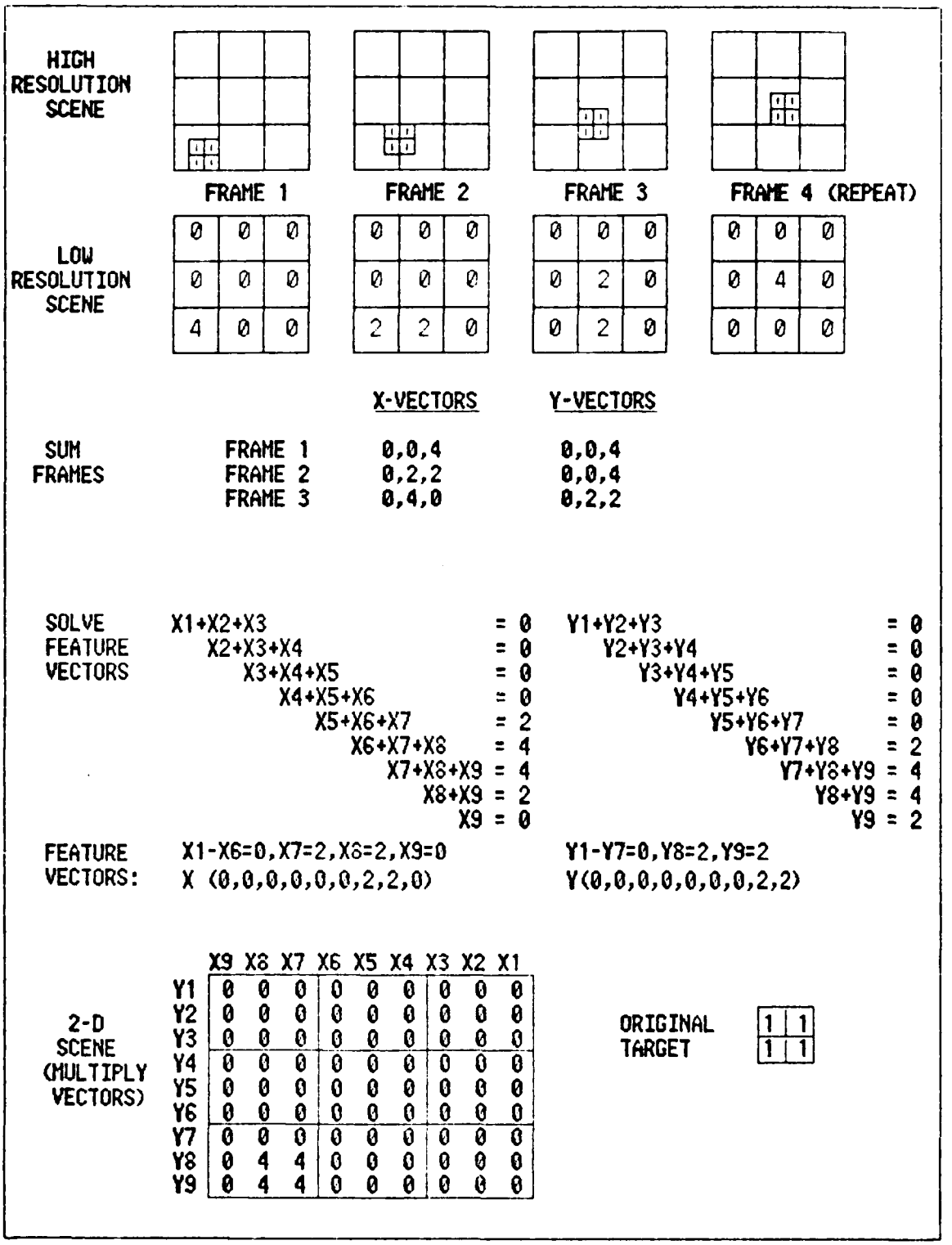


Figure 3-3. Derivation of Feature Vectors

detectors in four frames (the fourth frame is a repeat of the first.) The detector response for each frame is shown below the high resolution frames. By applying equation 18, the data are summed into vectors. These vectors form general equations which are solved simultaneously. The resulting $\bar{X}$ and $\bar{Y}$ vectors are the descriptors for the target. A two-dimensional general solution is obtained by multiplying the resulting vectors.

The two vectors $\bar{X}$ and $\bar{Y}$ derived from the above possess many desirable attributes to classify targets. Both vectors contain information on target size such as length and wing-span (within the resolution outlined in Chapter 2). Also included are the shape of the aircraft structure as defined by temperature contrasts. While these vectors are only generalized solutions to specific targets, they also represent projections of the target form which will be shown to be separable in representative signal-to-noise cases. These feature vectors will be the basis for feature extraction algorithms described in later chapters.

Visualization of these feature vectors as energy profiles of target signatures reinforces the usefulness of the feature set to describe characteristics of the target. The individual data points contained in a feature vector are actually an integration of the target signature over the area defined by the resolution width of the sampling process.

value of the subpixels to an element in either the X or Y vectors. By solving these equations simultaneously, solutions for the subpixel values can be obtained. This linear process reduces the X_i and Y_j vectors to single vectors $\bar{X}(l)$ and $\bar{Y}(l)$ (l is the length of the vector equal to NF_p.) These vectors are referred to in subsequent discussions as "feature vectors". The result of this manipulation is to increase the resolution of the extracted target signature by combining several frames of data into a higher-resolution one-dimensional vector. The equation for the subpixel solution generates either X or Y feature vectors and is given as:

$$\begin{bmatrix} h_{11} & h_{21} & h_{31} & \cdots & h_{k1} & 0 & 0 & 0 & \cdots & 0 \\ 0 & h_{21} & h_{31} & \cdots & h_{k1} & h_{12} & 0 & 0 & \cdots & 0 \\ 0 & 0 & h_{31} & \cdots & h_{k1} & h_{12} & h_{13} & 0 & \cdots & 0 \\ \vdots & & & & & & & & & \\ 0 & 0 & 0 & \cdots & h_{k1} & h_{12} & h_{22} & \cdots & h_{k2} & 0 \\ \vdots & & & & & & & & & \\ 0 & \cdots & & 0 & 0 & h_{N1} & h_{N2} & \cdots & & h_{Nk} \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} H_1(1) \\ H_1(2) \\ \vdots \\ H_1(k) \\ H_2(1) \\ H_2(2) \\ \vdots \\ H_N(k) \end{bmatrix}$$

where, (19)

$H_j(k)$: is the value of pixel j of frame k (either X_i or Y_j)

h_{ij} : is the subpixel i of pixel j.

A simplified application of the previous equations is illustrated in Figure 3-3. In this ideal example, the target consists of a square subdivided into four areas each with a value of one. The square moves across a 3 x 3 array of

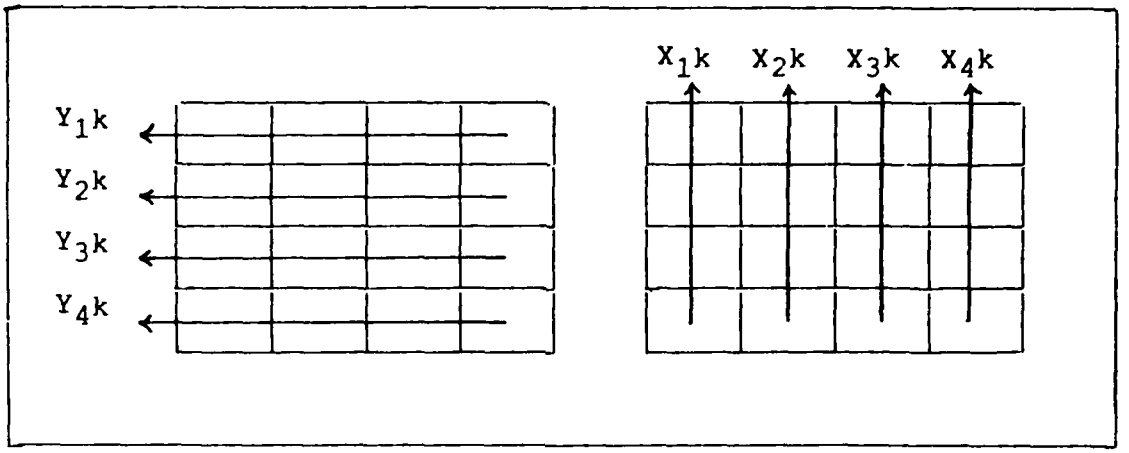


Figure3-2. Summation of Focal Plane Data to Obtain Sampled Data

Frames of focal plane data are therefore reduced to vectors and can be represented by,

$$X_ik = \sum_{j=1}^{N} S_{ij}(k)$$

and, (18)

$$Y_jk = \sum_{i=1}^{N} S_{ij}(k)$$

By acquiring k frames of data, an equal number of X_i and Y_j vectors are produced. Each element in the vector represents an energy projection the width of a single detector at frame k. Since the target moves across the detector in F_p frame samples, each of these samples represent a subpixel resolution R_i (Reference Equation (11)). Therefore, each vector forms a general equation relating the

focal plane. Therefore, tradeoffs may be necessary between the sampling time of the target, and constraints imposed by the system and scenario dependent noise and data requirements.

It would be desirable to resolve the target spatially such that a human observer could recognize the target by viewing a display of the enhanced target signature. However, resolving the target in a two-dimensional spatial plane is generally not possible due to the under-determined system that results from data acquisistion. To see this, consider a N by N detector focal plane sampled at a rate F_R, with the number of samples per pixel given by equation (10) as F_p. Each sample F_p can then be thought of as resolving the pixel to F_p square subpixels. To spatially resolve the target on the focal plane, $N^2F_p^2$ samples would then be required, thus, each detector would require F_p^2 samples. However, only F_p unique samples are generated. A modified spatial representation is therefore suggested that sacrifices some spatial features for the increased resolution offered by the multiple frames of data. This representation can be realized through summation of each frame of data orthogonally into two vectors with length NF_p, representing the energy projection along the the focal plane X-Y axis for that frame as illustrated in Figure 3-2. In this example, the signals from detectors of a 4x4 focal plane array are summed during each frame of sensor data.

sensor, one pixel and subpixel resolutions are also possible with large sensor footprints.

As the previous example illustrates, there is generally very little shape information apparent in single frames of data. Therefore, the target feature descriptors must utilize as much information as possible from a shaped-blob translating across the SFOV. This information is generally in the form of a differential grayscale based on signal interaction with the focal geometry. Analysis of the interaction between the target signal and the focal plane geometry is a key factor in deriving a useful feature descriptor.

Feature Derivation

In order to use shape information to classify a low-resolution translating target, the information must be "gathered" over several frames of data. In this manner, target shape information can be combined to obtain a signature with more information. The target, in essence, will represent a higher resolution shape than may be obtained from a single frame of data. The amount of increased resolution is limited, however, by the temporal, spatial and noise characteristics of a given scenario. The temporal limits are governed by equations (11) and (12) in chapter II, and the spatial limits by the geometry of the detector footprint and the resulting column-row configuration of the

achieved. Exploitation of this process is accomplished in the preprocessing phase of the feature extraction discussed in the following section.

The position and length of the window boundaries are based on the target velocity, crossing geometry, and the LET. By using these factors, each window size is limited to that required to sample the target. Also, window positioning can be set-up to sample the target for a minimum observation time allowing for LET, and hand-off error.

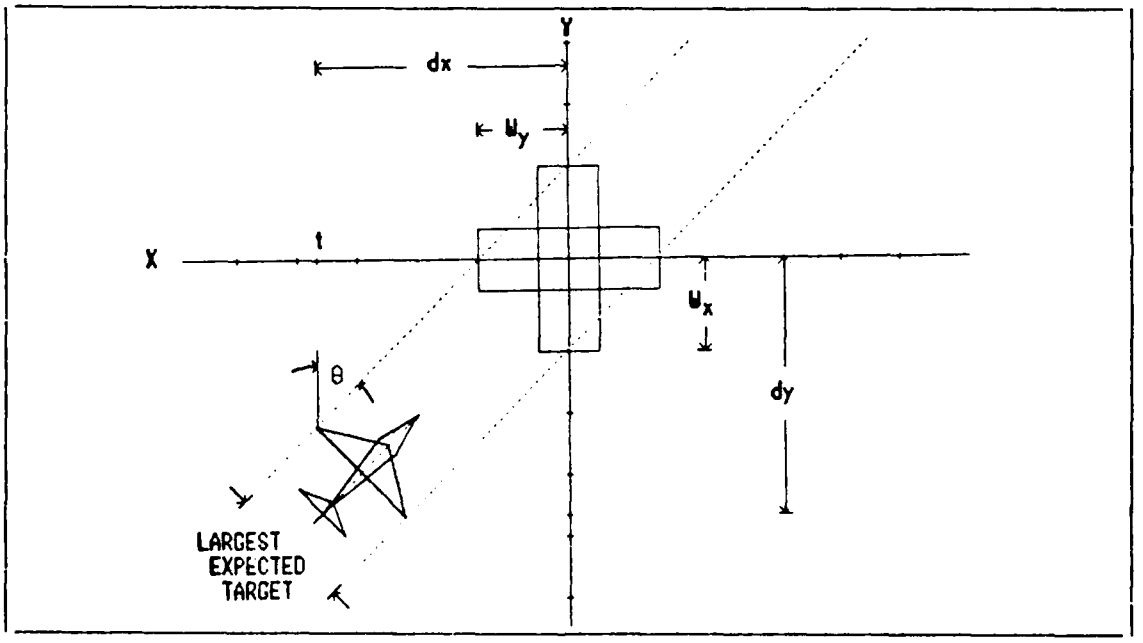


Figure 4-4. Target-Window Crossing Geometry.

The above illustration indicates the importance of crossing geometry and the LET in configuring the sample windows. The expected target trajectory is projected across the focal plane and the centers of the windows are then calculated. The following equations are used to generate the

sampling window parameters:

$$W_x = \frac{LET}{\sin\theta F_p} + W_e$$
$$W_y = \frac{LET}{\cos\theta F_p} + W_e \qquad (22)$$

and the window intercept:

$$dx = \frac{LET}{2F_p\cos\theta}$$
$$dy = \frac{LET}{2F_p\sin\theta} \qquad (23)$$

A typical configuration for redundant, limited sized window sampling is shown below in Figure 4-5.

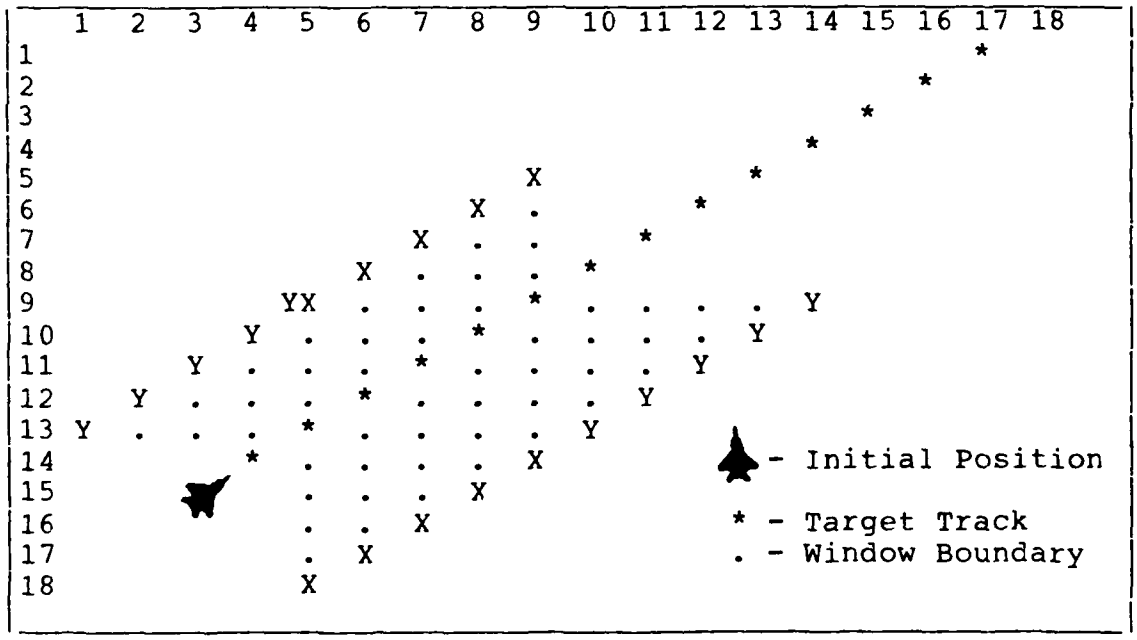


Figure 4-5. Redundant Window Sampling.

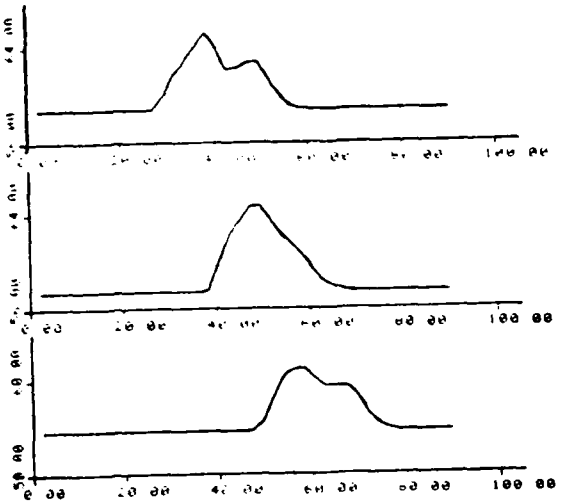
The use of window sampling does have limitations. Targets moving parallel to either focal plane axis will not

be sampled in that direction since there is little or no velocity component in the perpendicular direction. The target feature set is then reduced in half limiting potential classification. Fortunately, this occurs over a limited range target directions depending on the desirable lengthof the sampling window. For targets at grazing angles and limited backgrounds, the low velocity component may be an advantage in this case since longer detector sampling times are available.

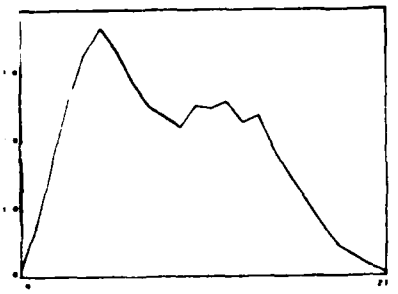
Vector Processing

Since window sampling consists of summing signals from several detectors during a time interval, the effects of drift, background spatial variations, and system induced degradations corrupt the original signal such that preprocessing is necessary before classification is possible. The preprocessing used to "clean" the feature vectors consists of two major steps; reduction of the data to a single set of orthogonal vectors, and deconvolution/filtering of known system effects such as the OTF and detector aperture functions. The vector processing proceedure described in the following text is illustrated in Figure 4-6.

After the target is extracted through all of the windows initially allocated, the background signal is then subtracted from the window data. If the sensor is stationary, this is a relatively simple procedure. If the sensor is moving, an estimate of the drift-induced signal on each window sample is

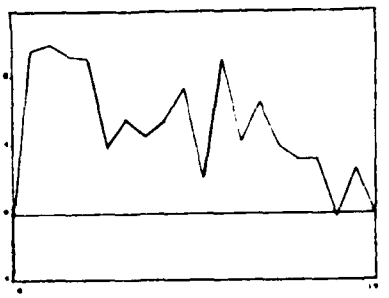


1. B-52 Target Signature Extracted From Parallel X Windows (250 Meters/Second, 20 Meter Res)

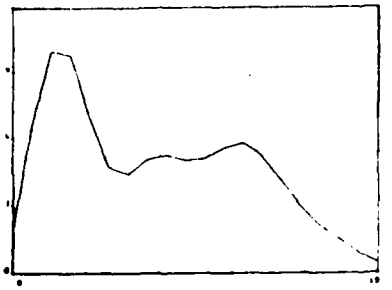


2. X Vector From Window Combination

3. Time-Domain Feature Vector After Deconvolution



4. Extracted Feature Vector After Low-Pass Vector



5. B-52 Model From Database (For Comparison)

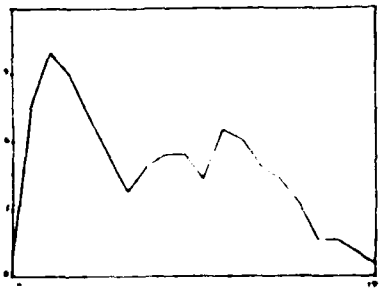


Figure 4-6. Vector Processing

derived. This is accomplished by sampling the drift signal both before and after target passage. If there is no drift, this signal will be DC. With sensor movement, the signal will be varying according to the net change in the background between samples. For slow variations, a least squares technique may be used to estimate the background signal during target passage. However, backgrounds with high spatial variations or high sensor drift may severely distort the target signature so that any useful data may be impossible to acquire. Unfortunately, the drift signal is approximately similar in frequency content to the target signal. Therefore, the target signal may be difficult to extract in severe drift-induced background noise.

Once the target signal is separated from the background, the windows are logically arranged to obtain the optimum target resolution. Windows are grouped into sets according to the phasing of the target with respect to the sensor as described in chapter II. For an in-phase target, each window would be a separate set with the number of sample windows equal to the number of sets. For an out-of-phase target which, for example, crosses two windows in an integer number of samples, each set is then composed of two windows with the number of sets equal to the total windows divided by two. The information as to the arrangement of the sets is also used in the deconvolution process as will be discussed in a later section. To reduce the logical window sets to a single

set, it is desirable to obtain the best target samples from each of the sets. This is done by reference to the noise history of each window. Those windows with low drift or DC noise are are averaged with corresponding windows in other sets. In this way, biases due to poor target-background contrast are averaged out. The remaining window set is actually a number of vectors to be combined into a single vector. The sequencing of the vector (from low coefficients to high) is a critical aspect of the vector recombination. There are several ways to order the vectors. One method is to assume the longest vector is first with gradually shorter vectors following subsequent. After ordering, the resulting vector is the generated by a round-robin extraction from each vector in the window set. Another method is to order the vectors according to the value of the first coefficient of each window vector in the set. A smaller coefficient indicates the energy in the vector may be spread out over a wider area. Once the vector is generated and normalized it is ready for deconvolution.

The deconvolution process attempts to remove both the finite window sampling blur and the optical blur. When the detectors comprising the windows are sampled, the detector aperture forces finite sampling window effects onto the data. The width of the detector window aperture is dependent on the number of samples (or frames) in which a point on the target is visible in the detector field of view.

Since the length of the window is represented digitally, it must be an integer value. Thus, pixel samples which are fractional use the window set approach outlined above to generate an integer length sampling window.

The optical transfer function (OTF) is modeled here as a two-dimensional Gaussian function. This function is separable as X and Y functions complimenting the windowing process. With the OTF modeled as a one-dimensional Gaussian, the transfer function of both the sampling window and the OTF is represented by the convolution of the OTF and the sampling window. This result is the deconvolution window used to recover the target signal. The deconvolution is performed in the Fourier domain for simplification and consists of the division of the spectrum of the target signal by the spectrum of the deconvolution window. This result is then low-pass filtered in the Fourier domain to remove high frequency noise induced from external sources and anomalies from the feature extraction process itself. The calculated target spectrum is then inverse transformed to obtain a time-domain target signal representation. A flow chart of the feature vector process is shown in Figure 4-7.

The feature extraction process has several advantages. First, the data can be acquired in real-time since no pre-processing is necessary to collect the data. Second, extraction of the data naturally decomposes two-dimensional functions into their X and Y components. This greatly

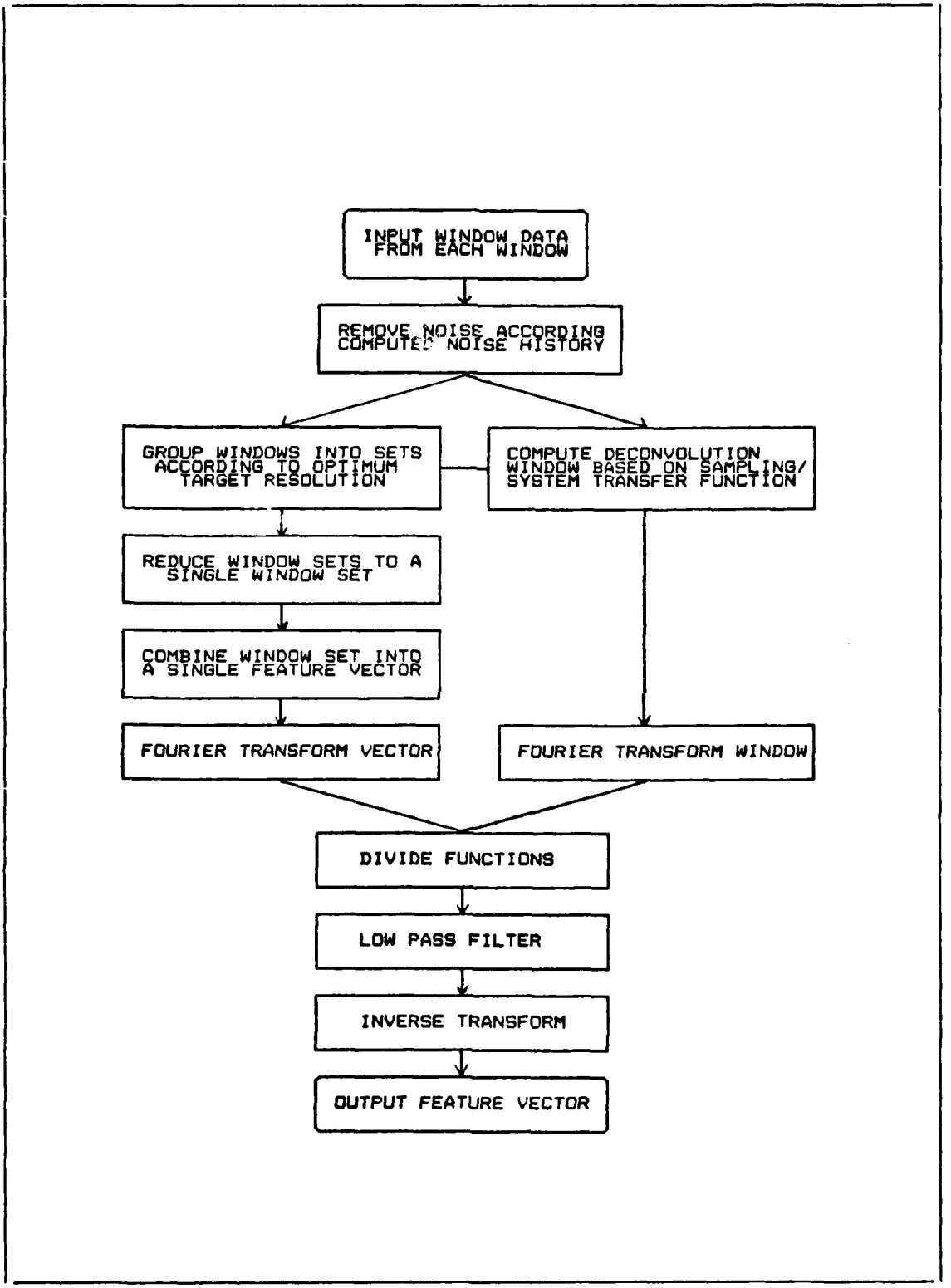


Figure 4-7. Feature Vector Processing Flow Chart

simplifies the deconvolution process. Third, memory requirements are minimal even with large numbers of windows. Finally, depending on the scenario, the necessary sampling or observation time may be 10 to 40 frames. This equates to 250 milliseconds to approximately 1 second for slowly moving targets.

Target Classification

The target feature vectors extracted from sensor data are compared to a database of target signatures for identification. Access to the database is controlled by the target orientation (determined from observed target motion relative to the X-Y focal plane geometry.) The success of the classification scheme is dependent on the method by which the potential target data is compared with that of the database. This study tests four methods to compare data. All of the techniques use some sort of normalization and/or interpolation to perform the comparison. For this discussion, model data will refer to signatures from the target database, and processed data will refer to the feature vectors of a potential target extracted from sensor data.

The target database is constructed from data taken from the five model aircraft discussed in chapter II. Each of the targets are digitized at a high resolution and at discrete orientations. The target data is then summed into rows and columns to form model feature vectors and saved in a database

file. To perform target classification, the input process data is compared to each of the database models according to the processed target orientation. The model which has the minimum deviation from the processed data is judged to be the processed target. Examples of the target data base are shown in Appendix A. Each target X-Y projection is shown at ten orientations from 0 to 90 degrees.

The methods used to compare the model and processed data evolved from the nature of the data itself. Recall that each point in the feature vector represents an integration over a area specified as the resolution of the data. To compare the high resolution database models to the low-resolution processed data, normalization is required in both magnitude and length for an accurate comparison. The techniques used to accomplish the comparison are a normalized polynomial interpolation, a matched integration, and a integrated linear interpolation.

The Polynomial Interpolation method uses the Newton form of the interpolating polynomial to obtain a polynomial fit of of model data whose order is equal to the length of the model vector. The method bases its comparison of the processed data to the model data entirely on the shape of the curve. (All data is normalized in magnitude and length prior to interpolation.) Disadvantages of this method are the erratic nature of the polynomial at higher orders, and the elimination of length as a feature.

The Matched Integration method performs a comparison by integrating the model data to match as closely as possible, the resolution of the processed data. To obtain the proper integration interval, the method tries to match the first point of the normalized processed data with successive integrations of the model data until the first points are relatively the same. Next, the remaining model data is integrated to match (as closely as possible) the resolution of the processed data and the data is then compared point-by-point. The difference between the model and processed data indicates the degree of dissimilarity.

The Integrated Linear Interpolation method combines the best features of the two previous methods. The model data is first integrated to match the resolution of the processed data. This consists of adding model data points to match a single processed data point and repeating. For example, given a model database with one meter resolution and a processed target which is extracted at a 2.25 meter resolution, the model data is integrated to generate a 2 meter model target (every two points are added together.) To correct for the resolution spacing difference between the model and processed targets (2.25 verses 2), the processed data is linearly interpolated at points defined by the model data. Therefore, the processed target will have data points at 2.25, 4.5, 7.75, etc. and the model target will have data points at 2, 4, 6, etc. meters. To find the distance

between targets for example, the processed data is interpolated between points 2.25 and 4.5, at point 4 given by the model data. Again, the difference between points (mean-square error) indicates the closeness of the data. An overview of this process is shown in Figure 4-8.

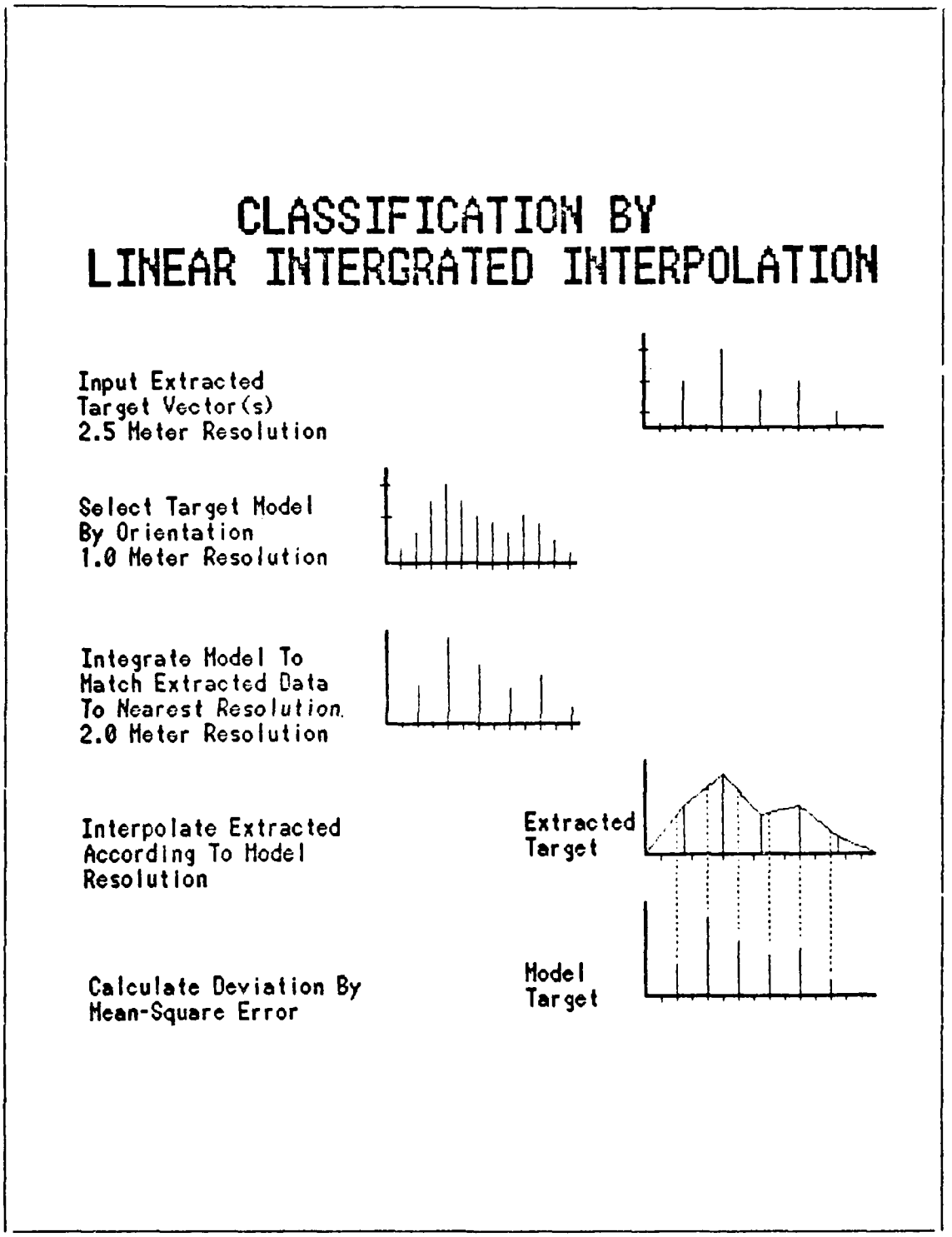


Figure 4-8. Overview of Linear Integrated Interpolation

V. Software Simulation Description

Introduction

To simulate data produced by a staring infrared sensor, and to provide a software testbed for classification algorithm development and test, a computer simulation was developed that incorporates key features of staring sensor performance. This chapter presents a description of the end-to-end software simulation used to implement and test algorithms discussed in previous chapters. The simulation is designed to allow user-tailoring of scenario dependent parameters and is useful for simulating various types of targets and backgrounds both statically and dynamically. The simulation also provides flexibility in analyzing algorithm performance at various points in the data processing path. An overview of the end-to-end simulation is shown below.

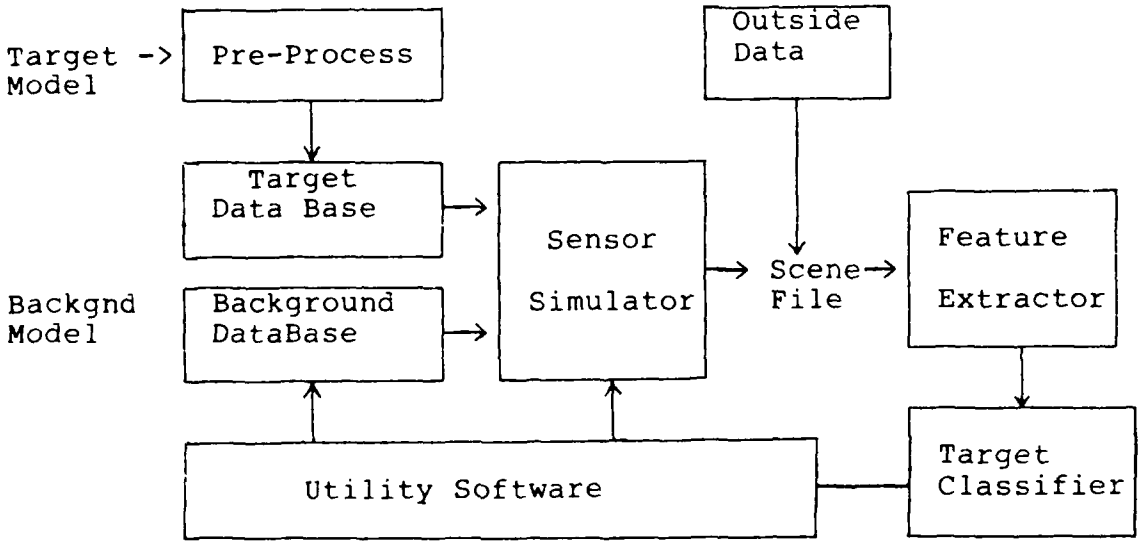


Figure 5-1. Software Simulation Overview

VI. Test Results

Introduction

Tests were conducted using the software simulation to determine the feasibility of target classification in a low-resolution scenario. It is not expected that the test cases presented here will fully characterize the extraction process, rather, it is the intent to obtain a general description of the algorithm performance by varying certain target and sensor parameters. Given this philosophy, limited testing was accomplished by generating scenes whose parameters were defined to test important aspects of the target classifier. This chapter will discuss the test conditions and resulting data with respect to the performance of the target classifier.

Scene Generation

For each test case, a scene sequence is generated consisting of 20 to 50 frames of sensor data depending on the target speed, sensor footprint, and extent of the focal plane. By sequential movement of the target (and sometimes sensor) and application of system transfer functions, scenes are generated according to user specified parameters. The quality of feature vector extracted from sensor data is dependent on the prevalent noise and background conditions, the sensor, and motion of the target. It is therefore desirable to characterize the classifier by varying

closer match of the respective target. The model database target are indicated on the horizontal axis. Since this is a optimum situation (no background clutter), these results can be used as a measure for the tests described in Chapter VI.

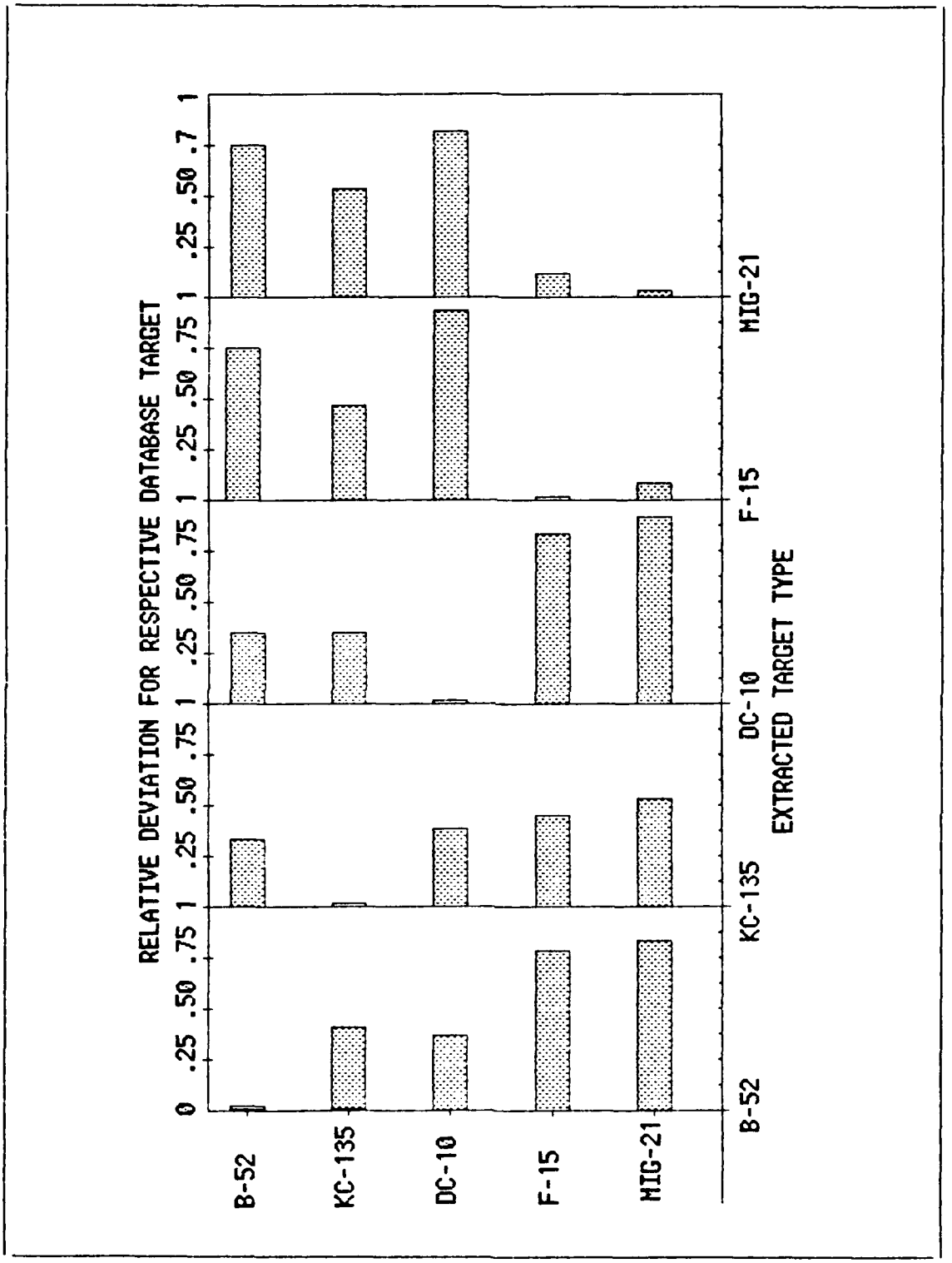


Figure 5-5. Targets Classified From Scenes Generated By the Scene Simulator (No Background)

are used to compare data, the results from routines INTERP and INTEGRATE are included for reference purposes only. Both these routines are superseded by the LINTEGRATE routine which performs an Integrated Linear Interpolation (see Chapter 4, page 54) to compare data. This method is less sensitive to scale and energy variations, and provides a more reliable comparison than either of the previous methods. The LINTEGRATE method was used to generate the data reported in Chapter VI.

The target database used by COMPARE is created by a utility software program call TARGEVAL for TARGet EVALuator. This program takes a 2-D target file generated from TGEN and calculates the associated target feature vectors for the target. The target feature vectors are then stored according to target orientation. This process is repeated for each target orientation from 0 to 45 degrees in 5 degree increments. The targets used for the data base have a 1 meter pixel resolution and are based on high resolution target images similar to larger target images shown in Figures 2-5 through 2-7.

To demonstrate the separability of targets in the target database, five scenes were generated using each of the targets without a background (zero energy.) The results are shown in Figure 5-5. The vertical axis indicates the deviation of the extracted target to each of the model targets in the database. A lower deviation indicates a

number samples per window set) and generating the associated window spectrum. The deconvolution is accomplished by dividing the spectrum of the input vector with the spectrum of the window. Following deconvolution, DECON filters high frequency components of the target spectrum by zeroing the high frequency terms in both the real and imaginary target data. The target spectrum is then inverse transformed and saved in a file for target classification.

Target Classifier

The target classifier is implemented in a program called COMPARE with three subroutines which perform comparisons according to the three methods described in chapter VI. The program begins by reading a processed target file created by PROFILE. After the data is normalized, COMPARE calls the three subroutines INTERP, INTEGRATE, and LINTEGRATE to compare the input data to each of the model target types contained in the target database. The model database is accessed according to the input target orientation. The results of each comparison are stored in a table and presented to the user at the conclusion of the program.

The results of COMPARE are given for each target type according to the deviation of the input target to each target in the database. COMPARE uses both X and Y target feature vectors if present in the input data. Although three methods

or columns of detectors) are summed and placed in a buffer. Since this is the only operation required during scene data processing, the feature extraction process continues at high speed with little processing overhead. It is important that the windows are setup far enough in advance of the target to obtain samples of the background prior to target passage. These background samples are also saved in a buffer to allow calculation of D.C. and drift induced background components for removal during vector processing.

Vector processing is accomplished by the profile subroutine VPROC. Currently VPROC separates only D.C. noise components from the windowed data. After removal of noise components, windows are combined and averaged according to the sampling resolution to be achieved (see chapter IV.) The data is then reduced to a single feature vector and is ready for deconvolution.

The sensor induced effects are removed with a inverse filter generated by prior knowledge of the OTF and the sampling window. The coefficients of the OTF function are generated from the routine GCOEFF (also resident in the scene simulator) and passed to the routine DECON which constructs the deconvolution window and manages the deconvolution process through calls to a one-dimensional fourier transform routine called "FOUREA". The deconvolution window is constructing by convolving the OTF function with the rectangular sampling window (whose length is equal to the

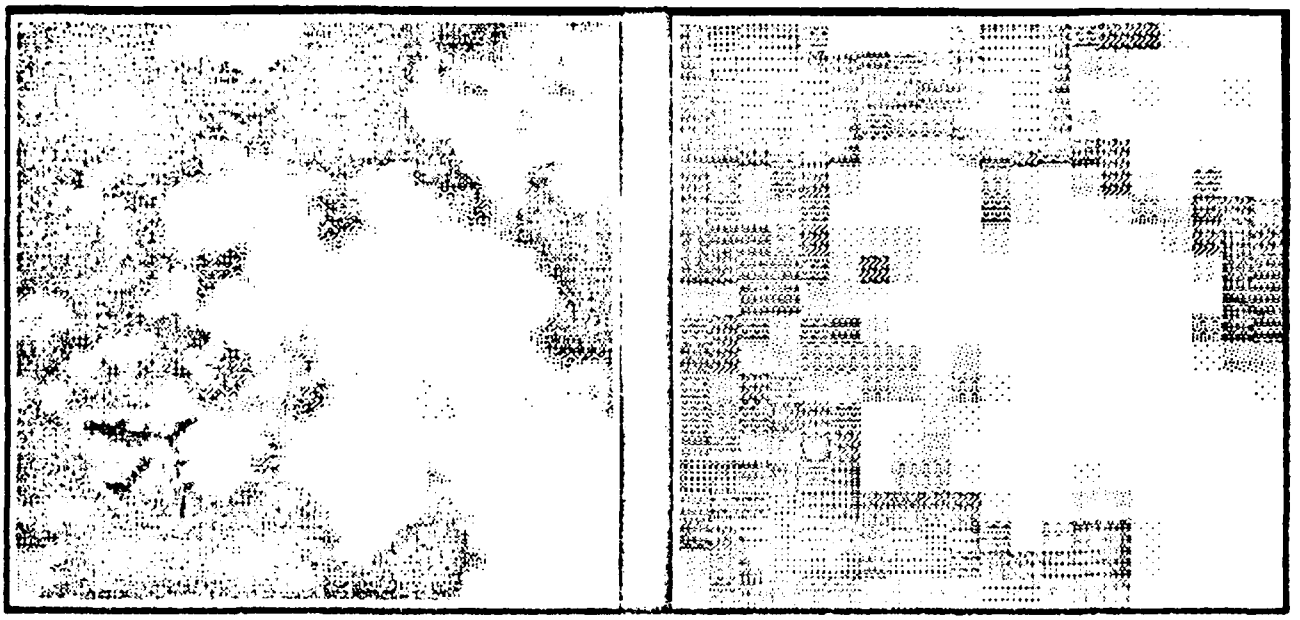


Figure 5-3. Sample Output of SCENE After Target Masking

Figure 5-4. Sample Output of SCENE after OTF.

detectors are possible. For simulations used to test the extraction algorithms, 10x10 and 20x20 detector focal planes were used to insure sufficient subpixel resolution.

A typical output of the scene simulator is shown in Figures 5-3 and 5-4. The high resolution scene after target masking is shown beside a low-resolution sensor focal plane output illustrating the averaging effect of the detector array. The resolution of the high resolution scene is 2 meters per pixel and the low resolution pixel is 10 meters. The next figure illustrates the same scene after addition of the gaussian optics blurring function.

Feature Extractor

The feature extraction algorithm described in chapter 4 is implemented in a program named "PROFILE". PROFILE can process targets moving in any direction (0 - 90 degrees) with minimal hand-off information from the target detection scheme. PROFILE currently only accepts data generated form the sensor simulation SCENE, however, any data which is preprocessed to include a file header may be used. There is no user-required information required to run the program since all needed information is contained in the scene file header.

PROFILE begins by calculating the target velocity vector and the sensor dependent parameters needed for window setup. After window initialization, data is processed frame by frame. The X and Y windows (consisting of contiguousrows

target data file. Thus, a target with a pixel resolution of 2 meters input into the scene simulator which is setup with a subpixel resolution of 15x15 defines the sensor footprint (L) to be 30 meters. The background used in this study is considered to have no scale limitations. Since the sensor footprint consists of an integer number of subpixels and is therefore not continuous, the selection of target speed and sensor motion is necessarily limited to those imposed by the subpixel resolution.

The background scene used extensively in this study consists of a photograph of a cloud layer as viewed from above. This scene is especially suitable for sensor evaluation since it contains both high and low spatial frequencies in an anisotropic pattern. Also, a target of any size can be superimposed without regard to the scale of the background.

Each scene file generating by SCENE contains a file header describing the conditions under which the scene file was generated. The number of frames generated by the scene simulation depends on the relative dwell time of the target and is usually specified so that the target crosses the entire sensor field of view during the simulation run. Focal plane sizes are user selectable ,however, the the number of detectors and associated subpixels are limited to the absolute size of the background scene after accounting for sensor motion. Thus, focal plane sizes from 10x10 to 100x100

through a window whose size is defined by the focal plane detector and sub-detector (pixel) dimensions. The window location may be fixed or variable depending of the user-selected drift rate parameter. Sensor-induced jitter may be modeled in a similar manner although is not implemented in the current version of SCENE. The target is then masked onto the background according to its current location in the scene sequence. If optics are included in the simulation, the OTF or point-spread function is calculated by convolving a averaging function whose coefficients are determined by a two-dimensional gaussian function generator implemented in a subroutine called GCOEFF. The OTF function is generated according to the pre-defined subpixel dimensions and the desired blur radius. For tests described in chapter six, a blur radius or variance of the gaussian function is approximately equal to the detector size for a matched-optics configuration. Next, the high resolution scene is integrated over the logical focal plane dimensions to model the detector averaging function. The mapping of the input high resolution data to the low resolution focal plane data is accomplished within the constraints imposed by the hardware. Each low resolution pixel is composed of an integer array of high resolution subpixels. The pixel dimensions are defined in odd subpixel sizes such as 5x5, 7x7, 15x15, etc. to allow easy computation of the optical PSF. The resolution of the subpixel is determined from the resolution of the input

version which sacrifices display capabilities for increased processing speed. User input data is shown in Table 5-1.

After the user enters preliminary information initializing the sensor system and target parameters, SCENE calculates sensor and target position and motion based on hardware pixel resolution. The program then enters a loop to generate frame sequences using the flow of operations outlined below in Figure 5-2.

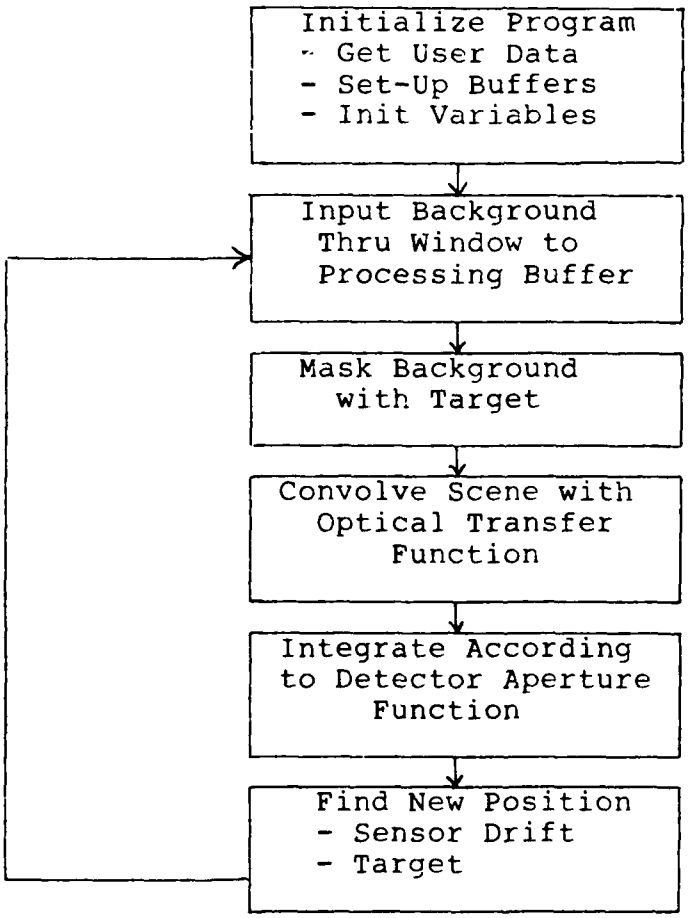


Figure 5-2. SCENE Data-Generation Flow-Chart.

The program reads the background data into a buffer

(Ref 10). This program allows storage of large data arrays accessible during program execution that could not otherwise fit into core memory (64K).

Sensor Simulation

The purpose of the sensor simulator is to generate scenes consisting of sequences or frames of data representing the output of a staring mosaic sensor. This process is essentially a integration of the input data (background and target) into low-resolution data as seen by an array of infrared detectors. The simulation includes key features such as specification of sensor geometry, optics, frame rate, and sensor motion. The sensor simulation is called "SCENE" and is implemented in two versions for use on different computers available in the lab; a Nova version which displays the generated scene data on a video monitor, and an Eclipse

Table 5-1.
SCENE User Input Data

User Input Files	Program Parameters
- Target (From TGEN)	- Focal Plane Size
- Background (Video File)	- Subpixel Size
	- Sensor Field-of-View (Window on Background Scene)
User Input Data	
- Frame Rate	
- Drift Rate	
- Optical Transfer Function	Output Data
- Blur Radius	- Filename
- Target Velocity	- Number of Frames to Generate
- Target Direction	
- Signature (Graylevel)	
- Initial Position	

target pixel resolution to be compatible with the sensor simulation. This reduced target represents the high resolution target as viewed by the staring sensor. The preprocessing program is called "TGEN" or Target GENerator. Before using the program, a picture of the target model is acquired with a TV camera and stored in memory of the Octek image analyzer. The field-of-view of the camera is determined by the scale of the plane model and is represented in meters. Thus, the user inputs two parameters to TGEN, the camera field-of-view (meters), and the desired pixel resolution (meters). TGEN will then reduce and separate the target to create the target file. The target file format is comprised of a file header and target data. The header information consists of the number of pixels that define the target, the direction the target is facing (0-90 degrees), and the pixel resolution. Data is stored by X, Y and intensity coordinates. The files created by TGEN are named according to the plane type, direction, and resolution to access the target database. The program uses the Octek to process the target and is not readily transportable.

Backgrounds used in the sensor simulation are digitized with the Octek and preprocessed to a 256 x 256 x 16 format for greater dynamic range in the scene simulator. The backgrounds are then stored as video files for access by the sensor simulator. Manipulation of video files is accomplished using a virtual memory utility called PICBUF

The simulation consists of five major components; the target/background preprocessor, the sensor simulation, the feature extractor, the target classifier, and the utility software. Tables and listings of software described in this chapter are contained in Appendix B. The utility software is dedicated to performing specific simulation performance measurements and maintenance of the simulation. This simulation configuration provides a flexible medium for algorithm evaluation whose results will be discussed in Chapter VI.

Target/Background Database

The targets and backgrounds used by the sensor simulator are generated using a video camera digitizer. The target database consists of five aircraft, a B-52, F-15, KC-135, DC-10, and a MIG-21. The plane models are 1:144 scale with the exception of the MIG-21 which is 1:72 scale. With the known scale, the actual size can then be represented in the simulation. The planes are painted flat-black to eliminate variations in intensity of the plane surface during digitization. Also, the plane resembles a relatively uniform silhouette at this point which is easily modified through software to represent a target signature of interest. The planes are digitized in a 256x256x4 pixel frame. This raw image is then preprocessed to generate the target file.

The target preprocessing consists of reducing the

parameters which define each of these components.

The sensor footprint is determined by the resolution of the target and the number of subpixels per detector. Since the overall size of the background scene is limited (256 x 256), the resulting focal plane size is likewise constrained. The choice of sensor footprint is therefore bounded by these restrictions and resulted in four sensor footprints chosen as sensor parameters for testing. The table below shows these footprints along with resulting focal plane and subpixel dimensions.

Table 6-1.
Sensor Footprints

Footprint Size	Focal Plane Dimensions	Subpixel Dimensions
10 meters	20 x 20 detectors	5 x 5 pixels
20	10 x 10	9 x 9
35	10 x 10	15 x 15
45	10 x 10	21 x 21

Other sensor parameters were selected based on the typical sensor of interest by FTD and by those typically used in other studies (Ref 1,18). Table 6-2 shows sensor specifications used to define the sensor model for the sensor simulation.

Detectors	Sample Rate
- Uniform Response - 8-12 micron - Background Limited	- Frame Rate 40 Hz
Optics	**Configuration**
- Gaussian Point Spread - Matched Optic/Detector - 100% Transmission	- Staring Mosaic - Nadir View

Table 6-2. Sensor Specifications.

A target is specified for a given test case by its type, direction, velocity and intensity.

Target Velocities were chosen according to subpixel resolution and vary over a range of 125 to 700 meters per second. Although several of the actual targets represented in the simulation are incapable of speeds in this range, the velocities are for characterization purpose only. The relative phasing of the target is controlled by the choice of target velocity. Most tests used both in-phase and out-of-phase velocities for characterization.

Target Direction is important in specifying the relative SFOV crossing geometry and the number and length of the feature vectors. For target directions from 15-75 degrees, two vectors are usually produced. For directions approximately parallel to the detector geometry, only one vector is produced. Intermediate directions within the 15-75

degree range result in changes in the X-Y velocity components and can be represented by either the velocity or direction parameters. Therefore, only two directions were used in the tests, 0 and 45 degrees. The 0 degree direction generates a single feature vector, and should result in a decrease in the classification ability of the algorithms. The 45 degree direction generates two vectors each of which has the same feature resolution.

Target Signal Intensity is specified by thresholding the target signature at a certain graylevel. This results in a relative increase in the signal from the target, and an increase in the contrast between the target and background. The contrast between the target and background is represented by a contrast ratio defined by the signal due to the background over the local area in which the target is observed, and the target signal itself. This value is analogous to signal-to-noise ratio and is intended to examine the performance of the target classifier over a range of target and background signal levels.

Test Cases

Trials were conducted at each footprint over a range of velocities and directions. The objectives of the test cases are oriented toward examining target separability with respect to different target-sensor scenarios. Specifically, trials were run to:

1. Determine the separation of target classes with decreasing information extracted for the target.

2. Determine the classifier sensitivity to orientation and velocity of targets.

3. Determine classifier performance with decreasing contrast or reduced target signal information.

4. Examine the effect of increased sensor footprint on target extractor performance.

5. Examine sensor motion influence on the feature extraction process.

To satisfy these test objectives, three categories of tests were performed. The first group of tests examined the effect of target motion on classifier performance relative to the SFOV at varying velocities and direction. This was repeated for each target and at the four footprints listed in Table 6-1. The second group of tests consisted of the five targets flown with an increasing contrast ratio. Each target traversed the SFOV at 45 degrees and at 250 meters/second. The third test involved drifting of the sensor during target observation. This was performed with the same target parameters as Group II. All tests used the cloud background discussed in Chapter 5 except Group II which uses a single graylevel background for several tests.

Table 6-3.
Summary of Tests

Test	Target	Vel	Dir	Contrast	Footprint
Group I	All	125 to 700 mps	0,45 deg	HI	11 meters 20 35 45
Group II	All	250	45	VARIABLE	20
Group III	B-52	250	45	HI	20

The pass/fail criteria for all of the tests are based on the target classifier results. The target classifier generates output for each test consisting of the relative deviation of the feature set under investigation and the target database. For targets that are successfully extracted, it is desirable to have a measure of the "confidence" of the target estimate. The measure used here consists of the interval between the correct target choice and the next closest choice as specified by the classifier. Therefore, an extracted target which compares similarly with two targets in the database should have a low confidence estimate, and a target that has no deviation from the corresponding database target while maintaining separation with other model targets in the database, should have a high confidence. A mathematical figure of merit called a

Confidence Distance is then defined as the separation between the correct target choice and the next possible choice given as a percentage:

$$\text{Confidence Distance}(\%) = \left[1 - \left(\frac{\text{Correct Target Dev}}{\text{Closest Target Dev}}\right) 2\right] \times 100$$

For Confidence Distances (CD):

< 1 : Target Classified Incorrectly
= 0 : Target does not separate
> 1 : Target Classified Correctly

Analysis of the relative "correctness" of the overall target extraction/classification will be in terms of the Confidence Distance above. Since there are actually three comparison methods used in the classification (Reference Chapter 4), the method which gave the most consistent results is used for the Confidence Distance measurement. This method is the Linear Integrated Interpolation which generated all of the test results reported here. The other methods did produce greater target separability for some tests, however, the Linear Integrated Interpolation method was by far the most reliable.

Results of Group I Tests

Approximately 175 trials were conducted (limited by computer time) with various sensor and target parameters defined for the Group I tests. In all but four of the tests the target was successfully extracted and classified from the

simulated data. The four tests that failed were high speed tests (greater than 600 meters/second) which resulted in only a small number of points defining the feature vector of the extracted target. However, even with an incorrect plane classification, it was still possible to separate plane classes. For example, a F-15 may be incorrectly classified as a MIG-21, or the KC-135 as a B-52. There were no tests which the separability of the target classes (transport versus fighter) was not maintained. With respect to individual aircraft, the larger aircraft separated with a higher confidence than smaller aircraft. This remained true at single pixel resolutions for each of the aircraft. This is mainly due to differences in wing structure between the large aircraft, and the larger number of samples that define the large aircraft. The results of a trial using a 20 meter footprint and out-of-phase target velocities of 250 meters/second is depicted in Figure 6-1. The figure can be compared with Figure 5-5 on page 68. Note the decrease in relative separation distance especially on the fighter aircraft. This is due to degradation effects of the background.

At higher target velocities, the resolution of target feature vector is limited to several data points. Therefore, the upper bound for the utility of the target classifier defined by the target speed with respect to the sensor.

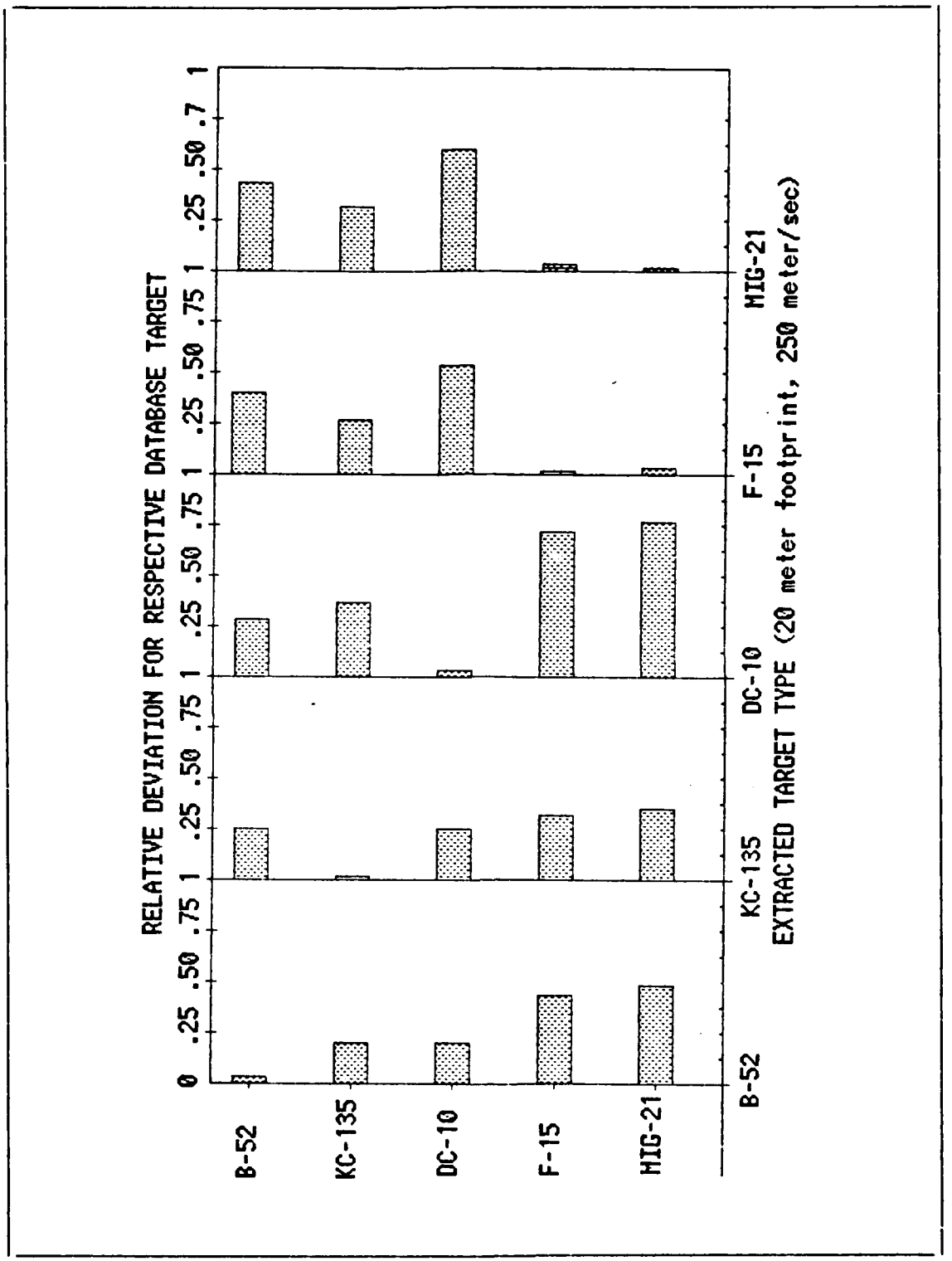


Figure 6-1. Results of Five Aircraft Extracted From a Cloud Background. (20 m footprint, 250 m/s)

sampling rate. For these tests, target speeds above 600 meters/second resulted in a rolloff of the classification confidence interval. These results are shown in Figure 6-2, where the five aircraft generated Confidence Distances over the velocity range indicated. At higher velocities, the relative phasing of the target becomes much more important. For out-of-phase targets, the combination of many frames of data results in higher resolution feature vectors and therefore, an improved Confidence Distance. This can be seen as a peak in the aircraft Confidence Distances at 460 meters/second in Figure 6-2. The five aircraft are shown over

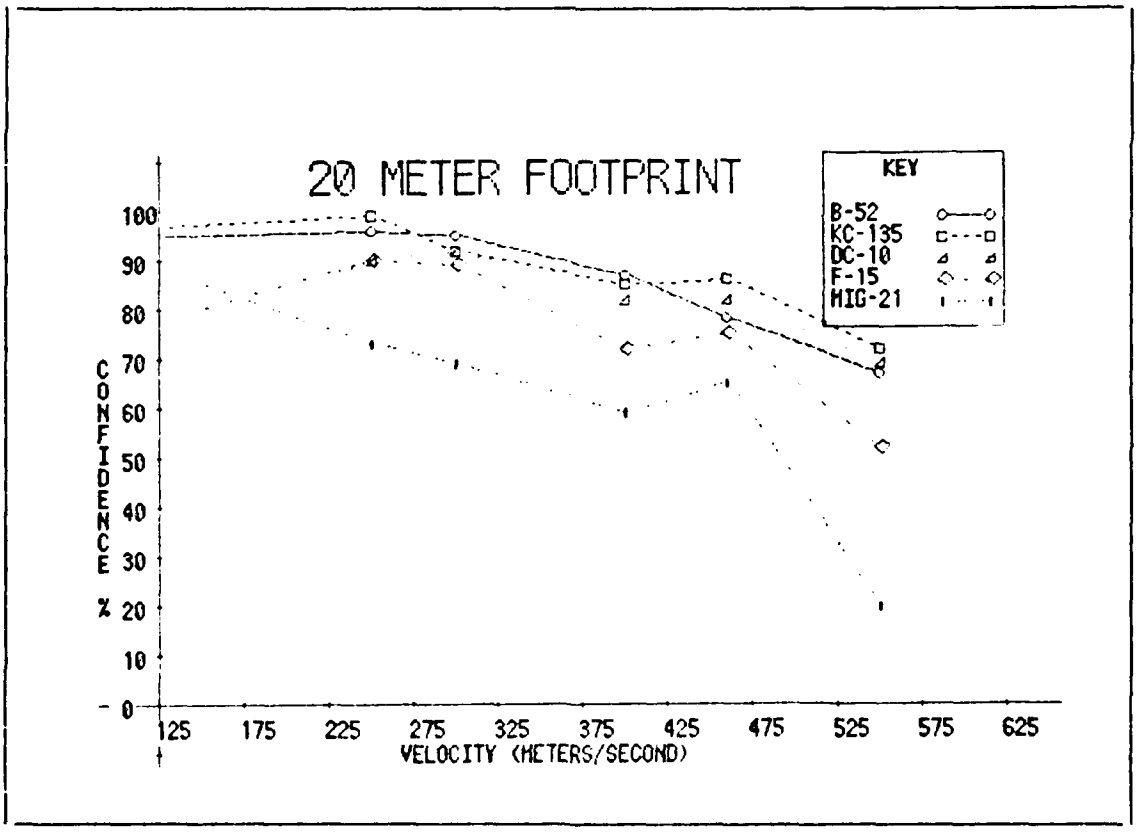


Figure 6-2. Rolloff of The Confidence Distances of Five Aircraft with Increasing Velocity

a range of velocities sampled with a 20 meter footprint. The effectiveness of the technique diminishes as the target moves toward an in-phase condition. Therefore, an in-phase target with a velocity of 600 m/s may be incorrectly classified, while correctly classified when out-of-phase at 650 m/s.

The target classifier was quite effective in comparing the extracted signature to the database model samples. The Linear Integrated Interpolation method provided the best results in comparing the data. The key features of the target which allow target separation are the wing structure and target length. The absolute energy in the extracted signatures was not a factor in separating the targets after the signature increased above a certain contrast threshold as discussed in the next section. Although it may seem that lengthalone may be sufficient to separate the aircraft used here, successful classification of extracted targets with resolutions of 5 meters demonstrate that wing shape is a dominate factor. Specifically, length indicates the difference between target classes (B-52 versus a F-15), and wing shape differentiates between intraclass aircraft (B-52 versus a KC-135).

From the analysis performed in Chapter II, the target velocity versus the system sampling function define the resolution of the sampled data. This would indicate that the sensor footprint resolution is not a major factor in defining the performance of the classifier. Therefore, the

performance of the classifier with a target at a 10 meter footprint should not be different than the performance with the same target at 50 meters. However, the system transfer functions and the associated vector processing seem to limit extrapolation of the test results to higher footprint sizes. Data from the tests indicate that there is a rolloff in the confidence intervals for all targets as the footprint size increases. An example of this is shown in Figure 6-3. This suggests that the increased losses from sensor transfer functions at higher footprints becomes more difficult to remove. At targets resolutions below one pixel, the blur or

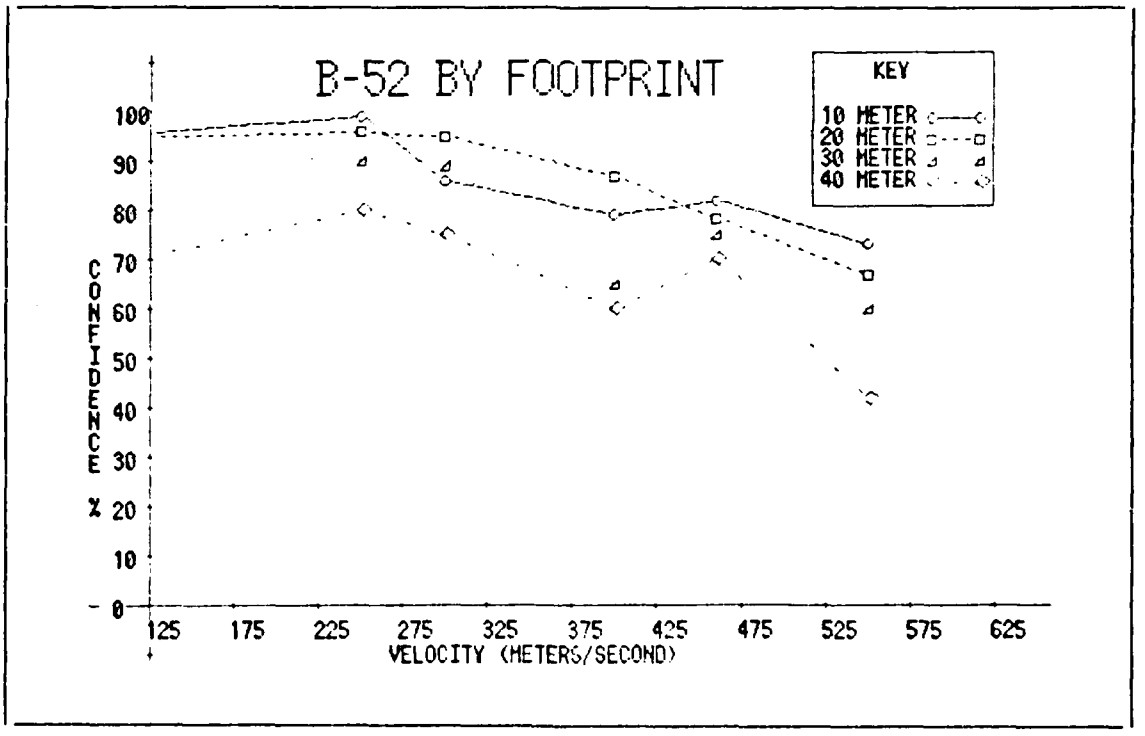


Figure 6-3. Rolloff of Confidence Distance Versus Footprint For aB-52 (velocity=250 meters/second against a cloud background)

low pass filter effect of the system reduces the information content of the target signature to a point were deconvolved becomes ineffective in recovering the target signal. In fact, for sensors with a large blur radius (DFOV< $_{\sigma}2$), the deconvolution process tends to distort the recovered time-domain feature vector. Another problem with extending this research to higher footprints is the limitation of dynamic range imposed by the system. The incremental energy difference from target motion across larger footprints is much less than that of smaller footprints with similar targets. Therefore, the target information is more difficult to extract and more susceptible to temporal noise in the respective DFOV. Therefore, from the results of these tests, an upper limit to the target classifier scene data is on the order of a single pixel on a target.

The results of the Group I tests indicate that the targets are separable within the velocities, directions, and resolutions tested. The performance of the classifier met expectations with relatively low error rates. The performance is bounded for decreasing input data resolutions and high target velocities.

Results of Group II Tests

The Group II tests measure the relative contrast between the target and background to determine the effect of

background clutter on target extraction. Two backgrounds are used in this test, the standard cloud scene, and a homogeneous background with a constant graylevel. The spatial distribution of the energy in the background cloud scene is anisotropic, so that both high and low target contrasts occur during target observation. This background energy distribution is particularly suitable to demonstrate the feature extractors ability to obtain a good target feature representation in a variable contrast scenario. The other background is created by filling memory with a constant graylevel value of 7 (on a scale from 0 to 15; 0 is Black, 15 is White.)

The first tests determined the ability of the feature extraction method to obtain good samples from targets with different energy contrasts. By increasing the target energy threshold, an improvement in average target/background contrast is achieved. Tests were conducted using the variable contrast cloud scene and the single contrast gray scene. Figure 6-4 shows the results of two aircraft against the cloud background. With reference to Figure 6-4, a lower target deviation indicates a closer comparison with the respective database target. The target energy refers to the relative contrast between the target and background. The data shows a relatively constant deviation for targets over the contrast range. These test results illustrate the classifier's ability to separate aircraft in a variable

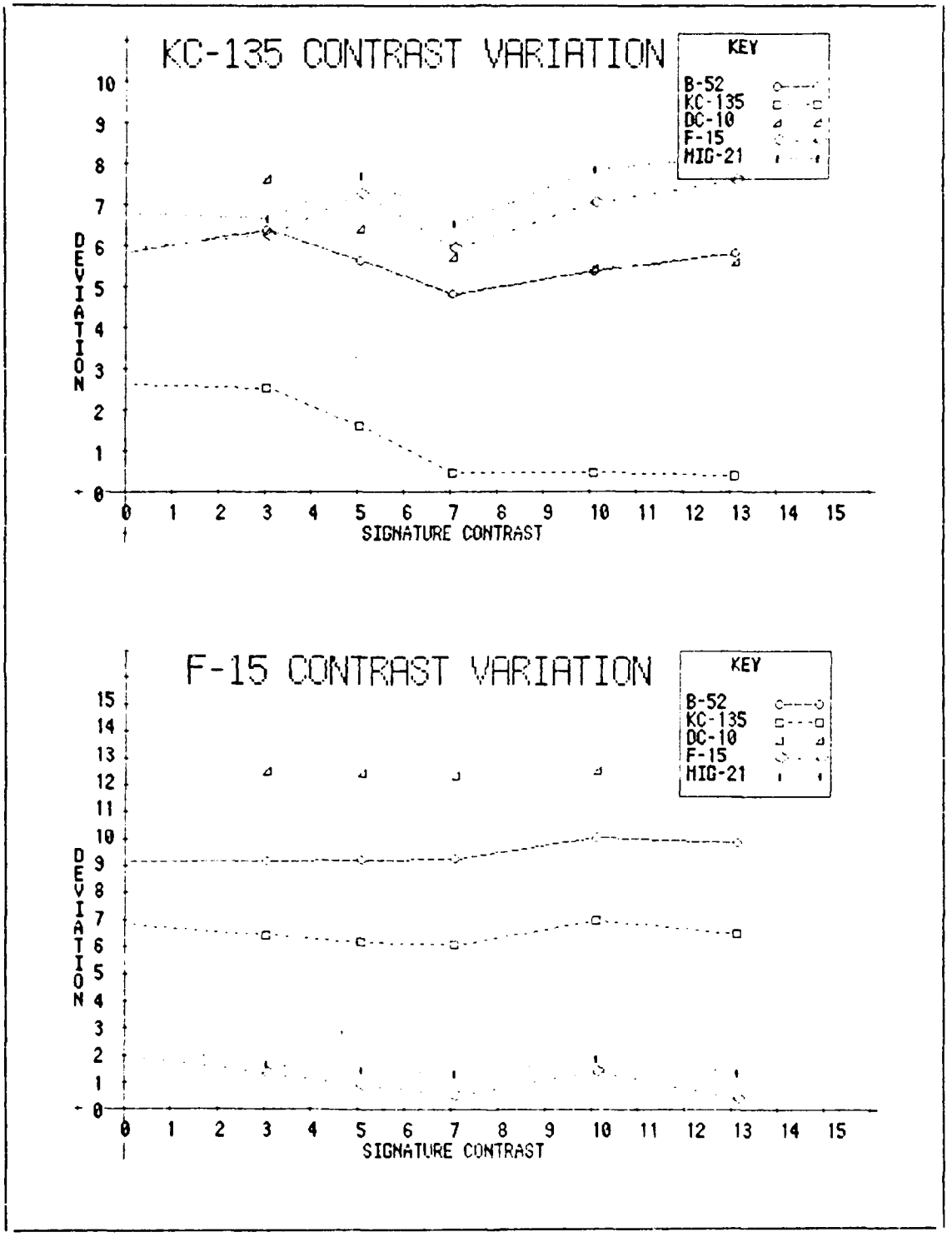
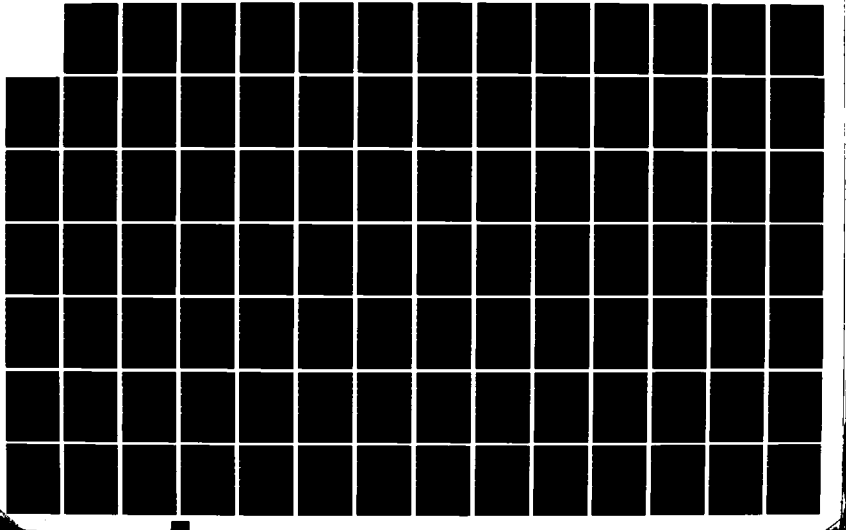
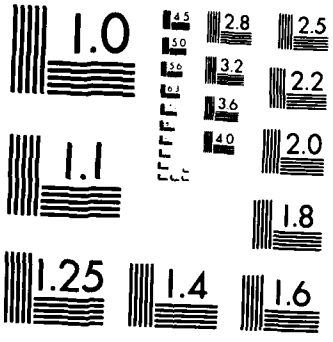


Figure 6-4. Effects of Variable Target Energy on Target Extraction Performance (Top:KC-135, Bot: F-15)

AD-A151 690 LOW-RESOLUTION TARGET CLASSIFICATION FROM A STARING INFRARED SENSOR(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING 2/2

UNCLASSIFIED J N TILLEY DEC 84 AFIT/GE/ENG/84D-67 F/G 17/5 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

contrast background. This analysis indicates that the feature extraction method using redundant window sampling, can consistently obtain a good sample of the target signature given the opportunity to view portions of target at high contrast. Therefore, this method takes advantage of good contrast in selected windows to generate a higher quality feature vector then could otherwise be extracted through a single window. This is illustrated in Figure 6-5, where the confidence intervals for the scene sequence are shown given multi-window versus single window sampling using the cloud background. Note the Confidence Distance is much less for the B-52 sampled thru a single window.

The remaining tests used the single gray-level background versus the cloud background. A KC-135 using the same scene parameters as depicted in Figure 6-4, was simulated against the gray background. The results are shown in Figure 6-6.

A comparison between a gradually increasing KC-135 signature against both backgrounds (from Figures 6-4 and 6-6) is shown in Figure 6-7. As shown, the KC-135 with relatively lower target energy can be extracted with a higher Confidence Interval when viewed against a variable contrast background.

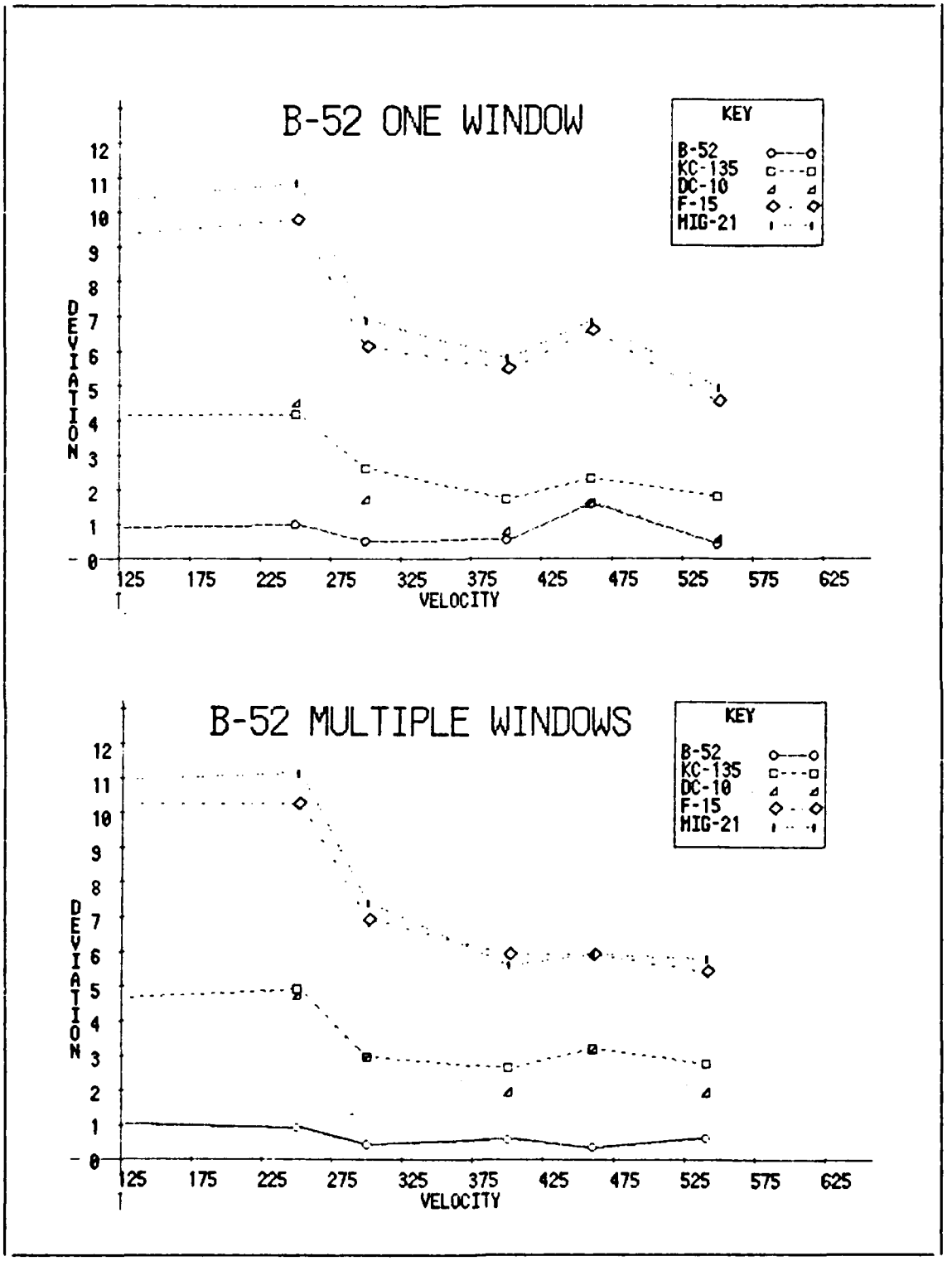


Figure 6-5. Comparison of Single versus Multiple Window Target Extraction (B-52 at 250 m/s)

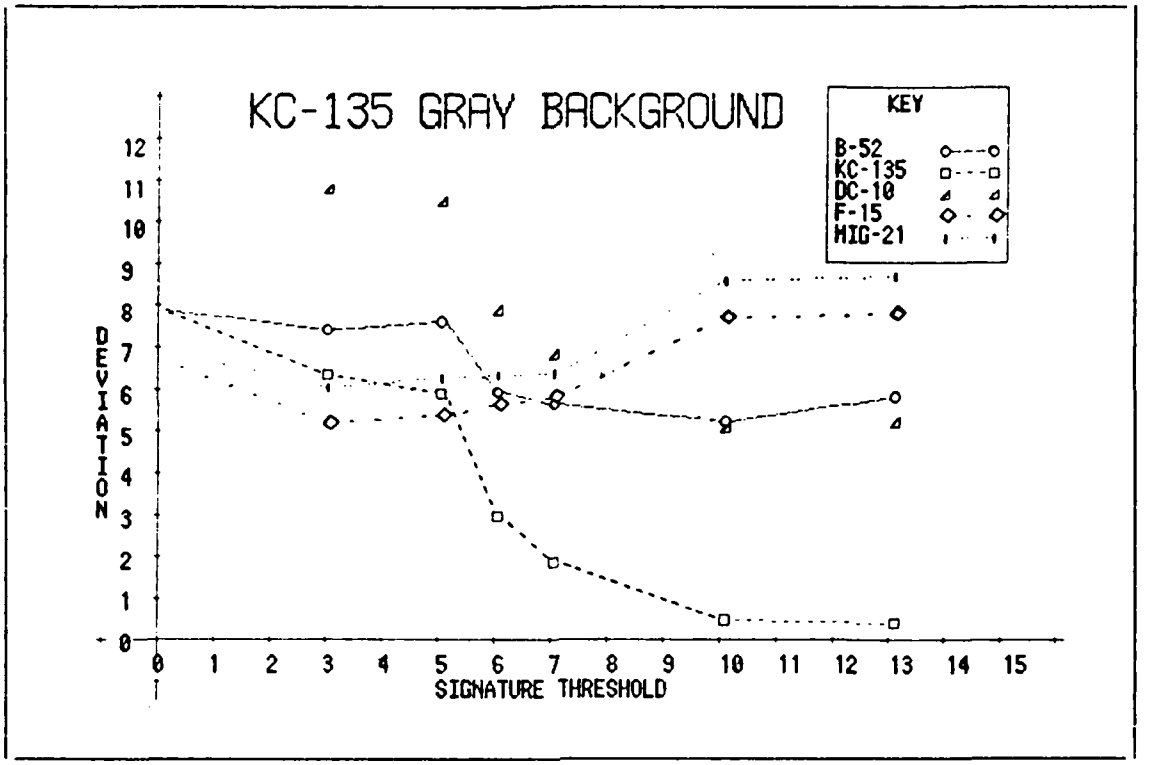


Figure 6-6. KC-135 Target Separation versus Signature Energy Against a Constant Background

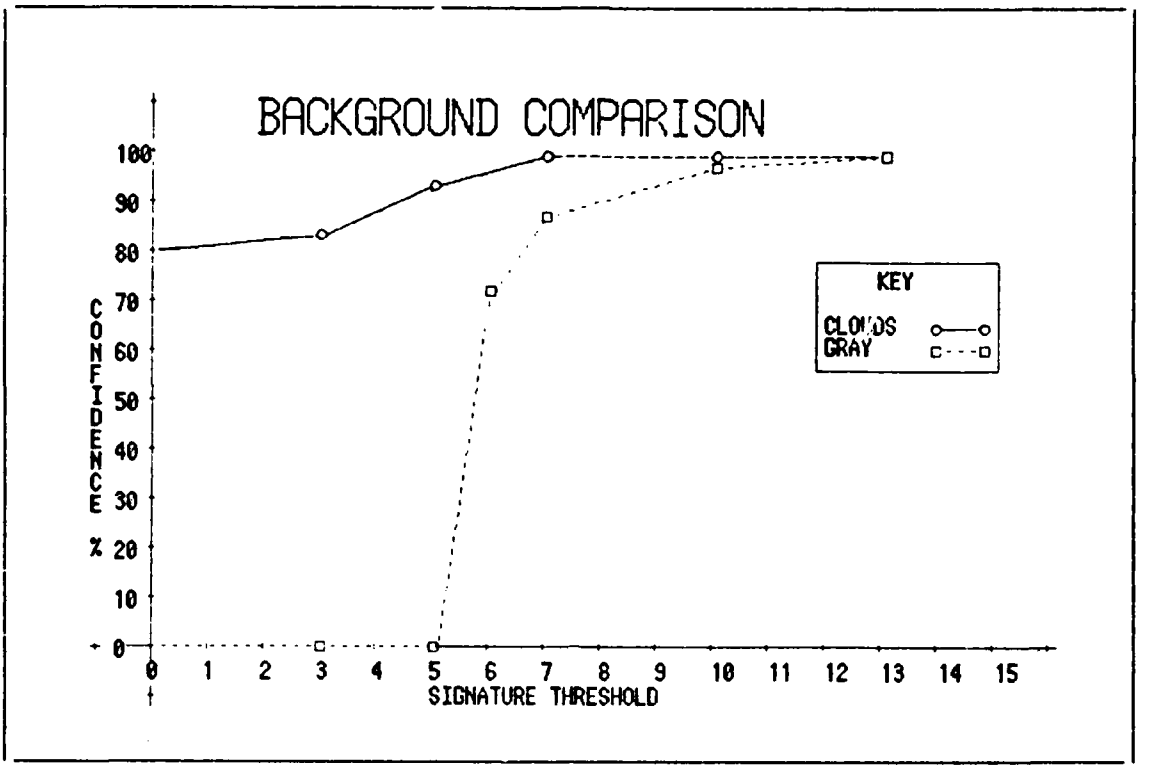


Figure 6-7. Comparison by Confidence Distance of a KC-135 Extracted From a Cloud versus a Constant Background

Group III Tests - Effects of Sensor Motion

The influence of sensor motion on target extraction is test using a 20 meter footprint and a sensor drift rate of 90 meters per second (or approximately 11% DFOV drift per frame). Figure 6-8 shows a comparison of data extracted from parallel sampling windows. The target is a B-52 with a velocity of 250 meters/second. The first set of curves a B-52 signature with no background, the other set of curves show a B-52 against a cloud background. Figure 6-9 shows the same target flying over the same background scene while the

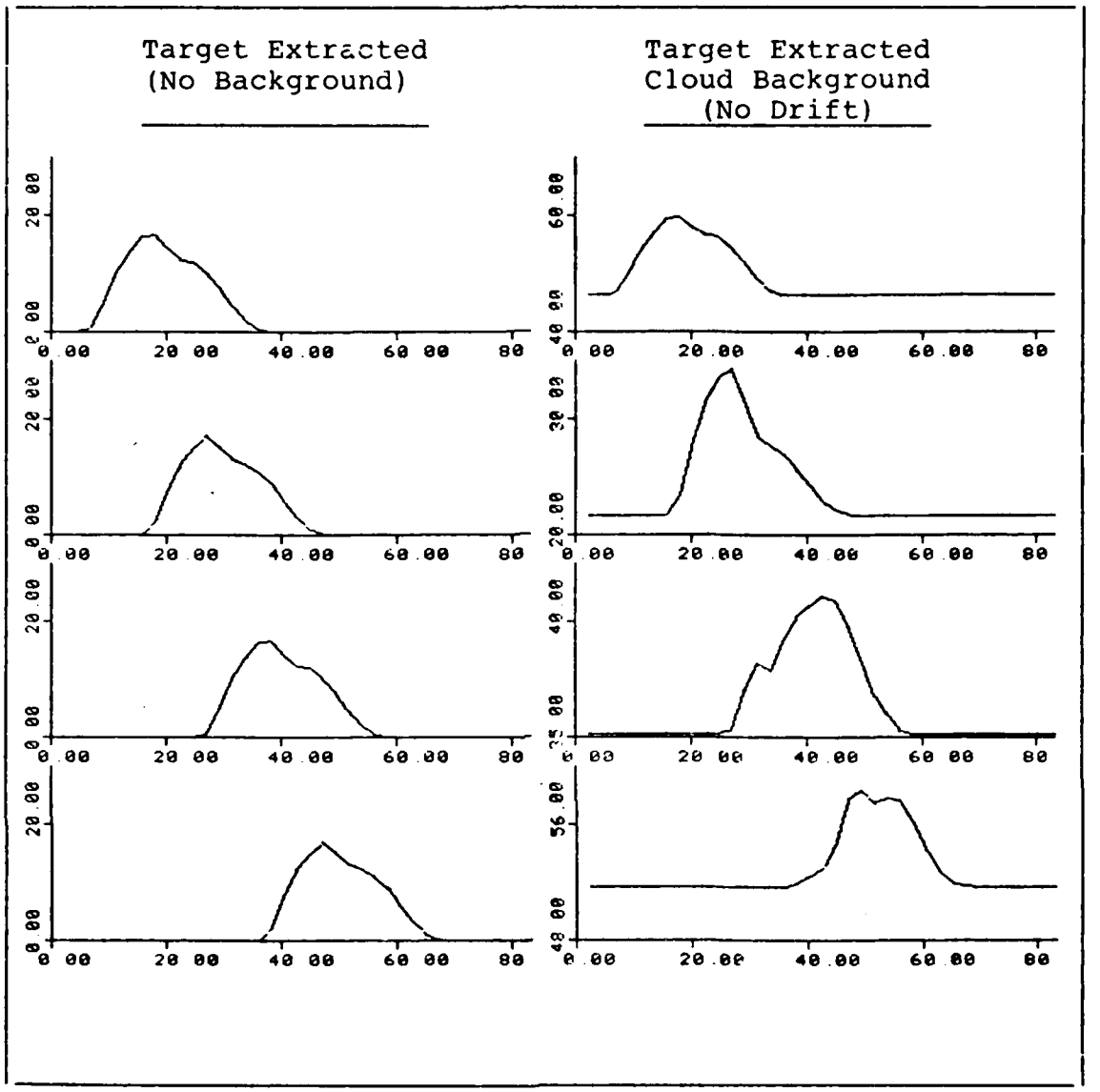


Figure 6-8. B-52 Signature Extracted from Four Parallel Windows (Left: No Background, Right: Cloud Background [no drift])

sensor is drifted. The severe degradation in the drifted data is apparent and the classifier failed to extractthe signature successfully. Unfortunately, the low frequency changes in the background when drifted are of the same

frequency and scale as the changes in signal due to target motion.

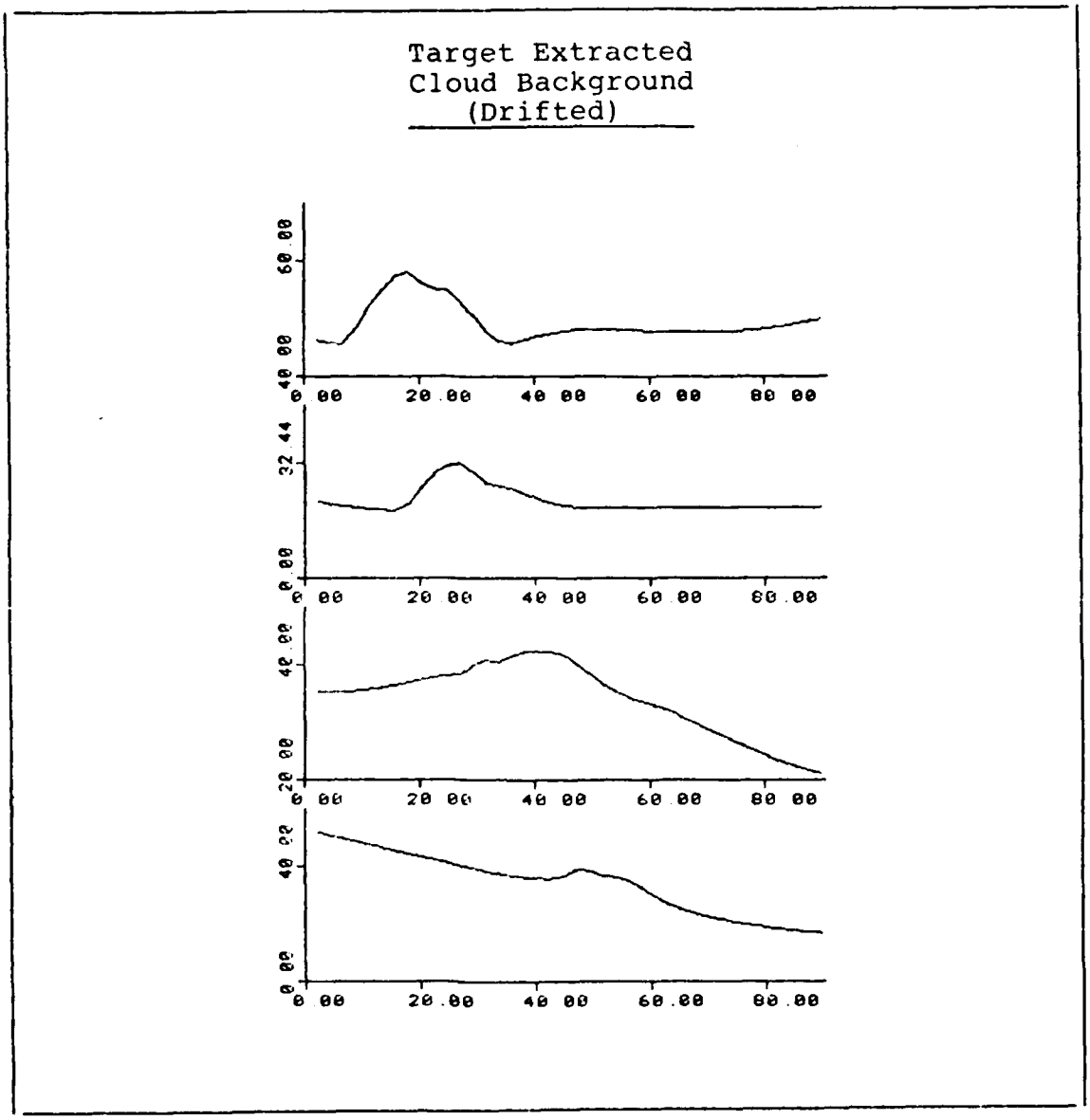


Figure 6-9. Effect of Sensor Drift on Target Signal Extracted From Parallel Sampling Windows (B-52 Against a Cloud Background)

The current target extraction algorithm demonstrates severe weakness in separating a moving background from a

moving target. The effects of sensor motion require further study before a practical application of the feature extraction method can be implemented.

VII. Conclusion And Recommendations

Conclusion

The methods developed in this study to extract and classify targets from simulated low-resolution staring infrared sensor data, are shown to be successful with various sensor and target scenarios. The classifier uses a feature set based on one-dimensional feature vectors extracted from sensor data. The feature vectors consist of orthogonol projections of the target spatial signature. The target information is extracted through logical sampling windows consisting of contiguous rows or columns of detectors. The windows are set-up prior to target passage based on minimal hand-off information from a target detection scheme. Increased resolution is accomplished by combining information from adjacent windows thereby taking advantage of the target's relative motion with respect to the sensor.

Results of performance tests indicate that aircraft are separable and can be classified with high confidence using the one-dimensional feature extraction method. Aircraft were successfully classified with single pixel resolutions. Extrapolation of the results of this study are limited by the sensor optics and sampling resolution. The loss of information from the sensor optics limit the size of the sensor footprint which the classifier can be applied (for a matched optic configuration.) Further limitations to

classifier performance depend on the target velocity and sensor sampling rate. The loss of target information due to poor target contrast can be offset by multiple looks if the background is anisotropic. Sensor motion generates severe degradations in the extracted data to the extent that classification is not possible without preprocessing.

The algorithms developed in this study can be implemented in real-time since little overhead processing is required. Target detection handoff information is limited to target position and direction. The extraction method can correct for errors in handoff information once the extraction process is initiated.

Recommendations

Since this effort consists of a target classification feasibility study which encompassed a broad range of topics, the depth of research was necessarily limited. Only limited analysis and testing could be accomplished due to time constraints. Therefore, recommendations for follow-on study consist of further analysis and testing with regard to understanding the physical mechanisms involved in loss of target information in a staring sensor scenario, and, the methods involved in countering these mechanisms to retrieve useful target information. One such method was presented here, developed in a "first-order" simplified sensor-model simulation. However, extending this study effort will

require enhanced models supported by solid analysis and testing. Therefore, specific recommendations are oriented toward understanding the staring sensor scenario, and development and test of enhanced extraction methods based on the improved sensor model:

1. Further research on thermal properties of typical infrared targets from space would support the assumption that a finite database could be generated for desirable targets. Since target graylevel can vary as a function of ambient operating conditions, analysis of target signature variations under different viewing conditions could establish if such a database is possible.

2. As shown in this study, sensor motion relative to the background can contribute severe distortion in the extracted target signal. By sampling the background around the target sampling window, and with knowledge of the sensor motion, it may be possible to estimate the motion induced signal variation. Further study is therefore required to determine the feasibility of separating background from target signals.

3. The sensor model used in this study approximated the transfer function for a staring infrared sensor. An improved sensor model could yield more conclusive results. This model should include an improved detector model to account for non-uniformity, finite detector response time,

detector integration time, and detector noise constraints.

4. To compliment three, the background scenes viewed by the sensor should include greater dynamic range and a variety of thermal possibilities (ocean, mountains, farmland.) Obtaining real infrared data could enhance the credibility of the test results.

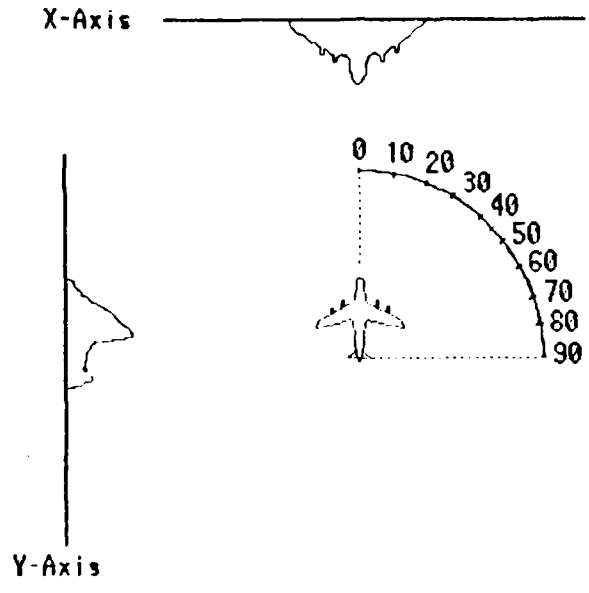
5. This study used the one-dimensional feature vector to describe and separate aircraft from IR data. However, given a larger number of target possibilities, [illegible]ture vector separation may be more difficult. Therefore, extensive testing using a large number of aircraft will be required to determine if the general solutions afforded by the one-dimensional feature vectors described in this study will separate aircraft.

6. It would be desirable to augment the target extractor software with a target detection algorithm so that data from different sources could be tested automatically without user interaction.

Appendix A

Target Database

The target database used to classify extracted targets is depicted in Figures A-1 through A-10 on the following pages. The five targets are represented as X and Y projections, each having 10 different orientations with respect to the vertical axis. The vertical axis (top of the page) is designated as zero degrees with angles increasing clockwise to the vertical axis. The diagram below illustrates the convention used to represent the data:



Each figure contains 10 signature graphs which represent the target signature projected against the indicated axis. The signature graph uses normalized energy (vertical axis), verses samples of the target defined in one meter increments (horizontal axis).

```
C*************************************************************************
C
C       PROGRAM COMPARE                    VERSION 1.1
C
C       FUNCTION: THIS PROGRAM COMPARES A PROCESSED X-Y VECTOR
C                 GENERATED BY THE RESOLUTION ENHANCEMENT ROUTINES
C                 AGAINST A DATA BASE OF KNOWN TARGET MODELS.
C
C       Macro:    FORT COMPARE
C                 RLDR COMPARE INTERP INTEGRATE LINTEG FORT.LB
C
C       LANGUAGE: FORTRAN IV
C
C*************************************************************************

        PARAMETER ASIZE=100

        REAL RXMAX,RYMAX,RXDIS,RYDIV
        INTEGER NUMPLANE

        REAL XPROC(ASIZE),YPROC(ASIZE),XPINDEX(ASIZE),YPINDEX(ASIZE)
        REAL XMODEL(ASIZE),YMODEL(ASIZE),XMINDEX(ASIZE),YMINDEX(ASIZE)
        REAL XPTEST(ASIZE),YPTEST(ASIZE),XPIN(ASIZE),YPIN(ASIZE),MPS
        DIMENSION INFILE(20),MODFILE(7)

C       INITIALIZE VARIABLES

        RXMAX=0
        RYMAX=0
        XPMAX=0
        YPMAX=0
        RXDIV=0
        RYDIV=0
        NUMPLANES=5
        ITEST=0

        ACCEPT"RUN TEST? ",ITEST
                IF (ITEST.NE.1) GOTO 14
                TYPE
                TYPE"   (1)   INTERPOLATION"
                TYPE"   (2)   INTEGRATION"
                TYPE"   (3)   LINEAR INTERPOLATION"
                TYPE
                ACCEPT"WHICH ONE? ",ITEST

C       GET PROCESSED DATA

14      ACCEPT"DO YOU WANT A PRINTOUT OF RESULTS? ",IPR
        TYPE "ENTER PROCESSED DATA FILENAME: "

        READ(11,16) INFILE(1)
16      FORMAT(S20)
```

LINTEG.FR	S,A	Calculates Deviation Between Target Vectors w/Linear Interpolation	Feature Vectors	Target Dev
INTEGRATE	S,A	Calculates Deviation Between Target Vectors w/Normalized Integration	Feature Vectors	Target Dev
INTERP.FR	S,A	Calculates Deviation Between Target Vectors w/Polynomial Interpolate	Feature Vectors	Target Dev

Directory Of Appendix B

GCOEFF.FR	S,A	Generates 2-D Gaussian Coefficients	-	2-D Array

Target Extraction

Program	Type	Function	Inputs	Output
PROFILE.FR	M,A	Extracts Target Signature Sensor Data	IR Data	Target Feature Vectors
DECON.FR	S,A	Filters Feature Vectors In Fourier Domain	X-Y Vectors	Feature Vectors
VPRROC.FR	S,A	Reduces Sampled Window Data to X-Y Vectors	Window Data	X-Y Vectors
PLOT10.FR	S,A	Plots Vector Data Via Calls to CALCOMP	-	-
FOUREA.FR	S,A	Computes 1-D Fourier Transform	Time Freq	Freq Time
GETHDR.FR	S,A	Reads Header From Sensor Data File	-	Header Data
GETDA.FR	S,A	Reads A Frame of Sensor Data From File	-	IR Data Frame
DDATA.FR	S,A	Displays Frames of Low-Res Data On Octek	IR Data Frame	-
PIXFRAME.FR	S,A	Calculates Target Velocity in Pixels/Frame	Target Parameters	-
FRAC.FR	S,A	Calculates Out-of-Phase Resolution	Target Parameters	-
FRDUMP.FR	M,U	Dumps IR Sensor Data Created By SCENE at Line Printer	SCENE Data	-

Target Classification

Program	Type	Function	Input	Output
COMPARE.FR	M,A	Classifys Extracted Targets With Database	Feature Vectors	Target Dev

Appendix B

Software Listings

The software used in this thesis is grouped according to application. This includes target/background generation, scene simulation, target extraction, and target classification. Within these categories the software is typed according to hierarchy and function. An application (A) or a utility (U) program may be a main routine (M), or a subroutine (S). The Software is written in FORTRAN IV and FORTRAN V for use on the Data General Nova and Eclipse computers respectively.

Target/Background Generation

Program	Type	Function	Input	Output
TGEN.FR	M,A	Generates Target Files Form Video Data	-	Target Files
TARGEVAL.FR	M,U	Generates Target X-Y Vectors		
INVPLOT	M,U	Prints Target Database At Line Printer	Target Files	-
PIXMOD.FR	M,U	Converts Standard Video Files to 16 Bit	256x256x4 Video	256x256 x16

Scene Generatio

Program	Type	Function	Input	Output
SCENE.FR	M,A	Generates Simulated Focal Plane Output	Backgrnd Target	Sensor Data
PSPREAD.FR	S,A	Convolves 2-D Video With 2-D Coefficents For Point-Spread Funct	Video File & PSF Data	Blured Scene

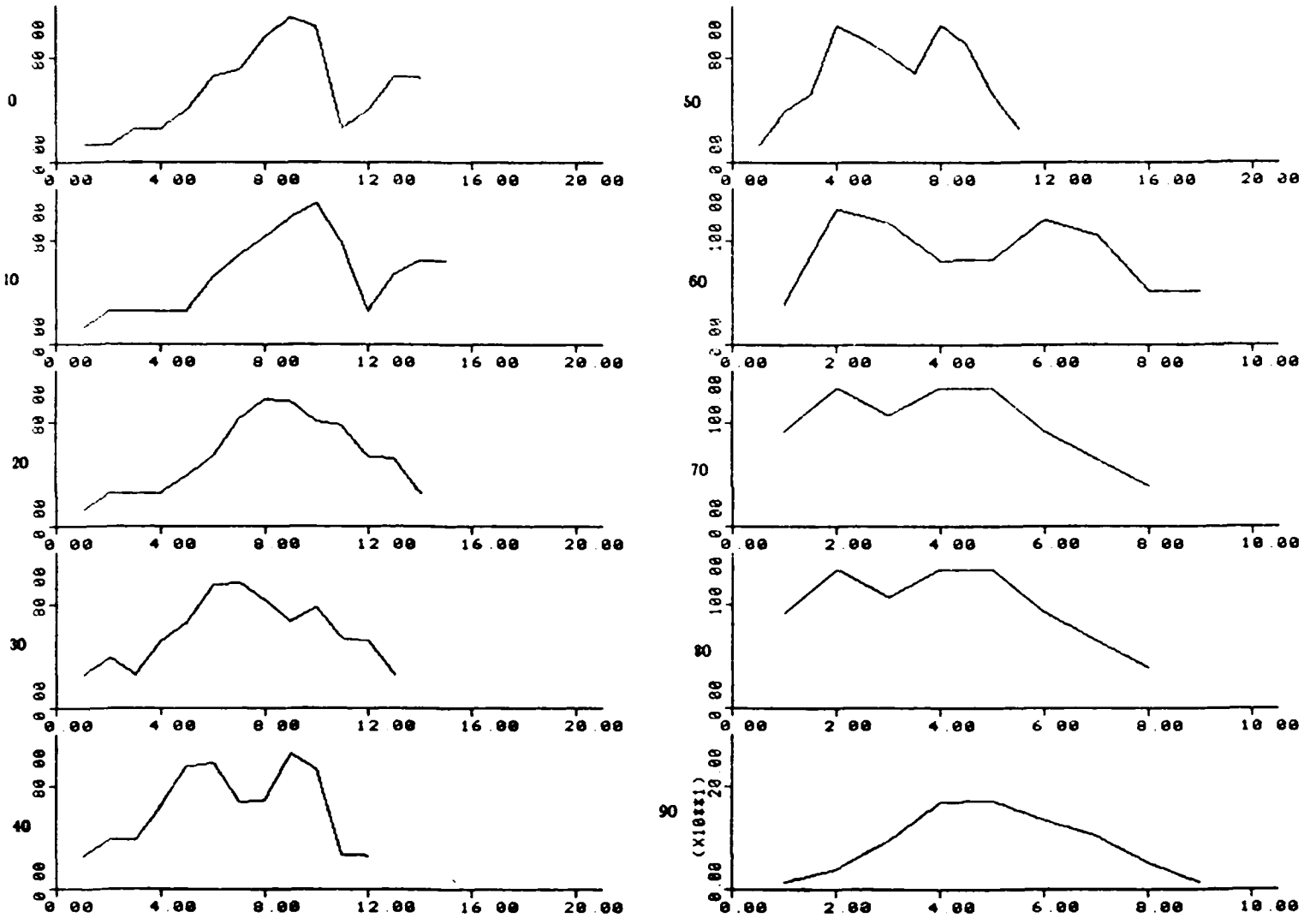


Figure A-10. Y-Axis MIG-21 Data at 10 Degree Increments

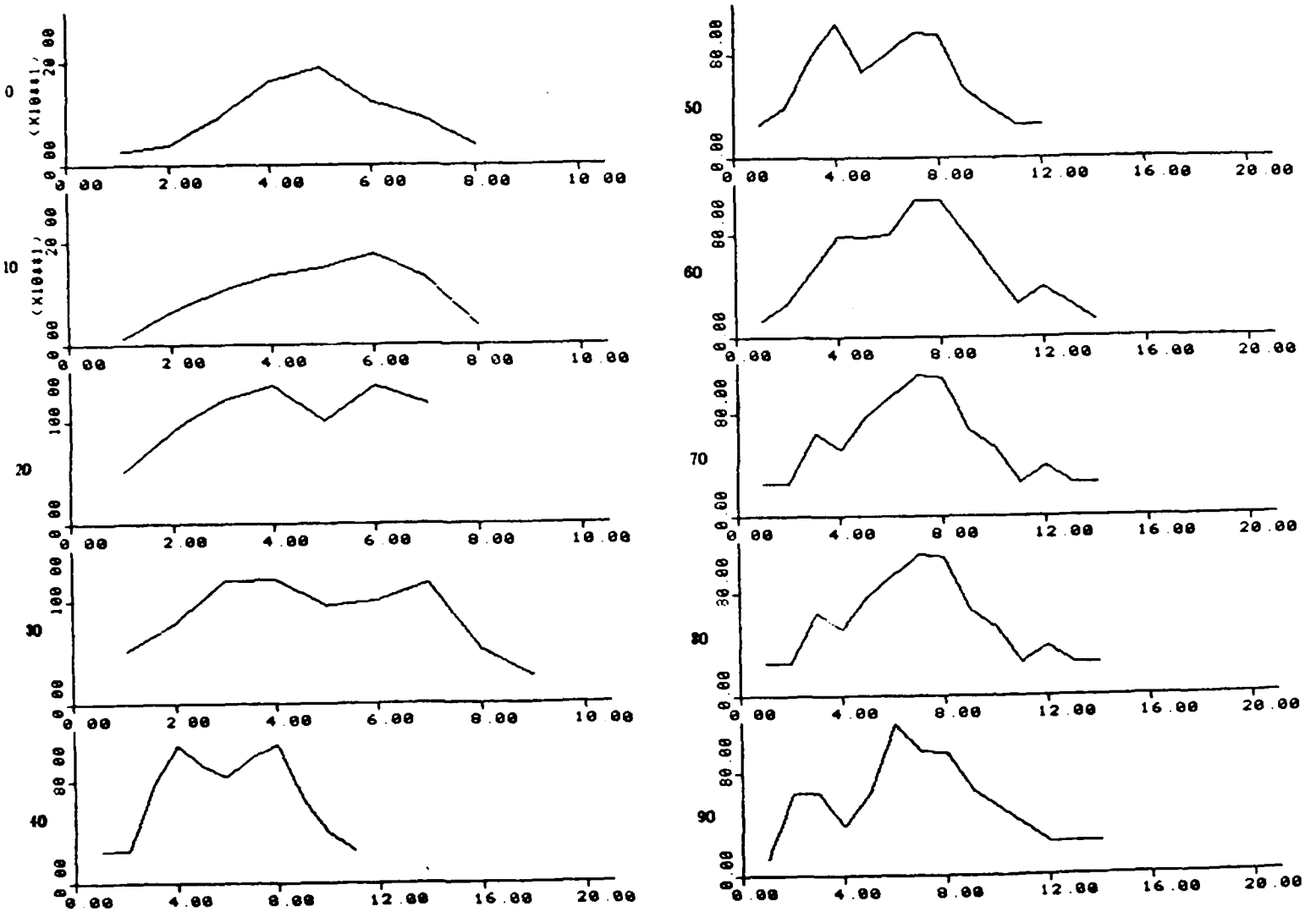


Figure A-9. X-Axis MIG-21 Data at 10 Degree Increments

F-15
Y-AXIS

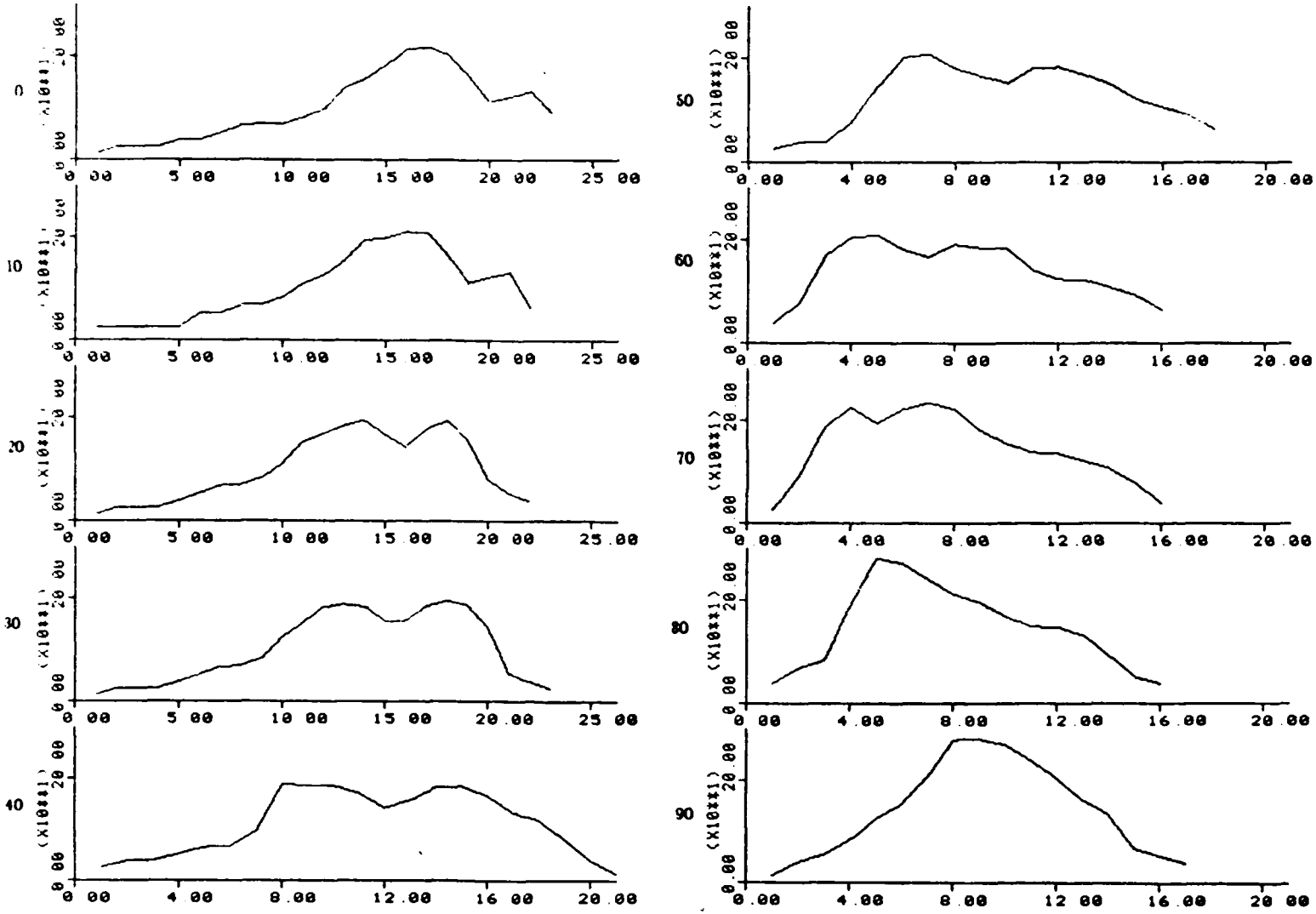


Figure A-8. Y-Axis F-15 Data at 10 Degree Increments

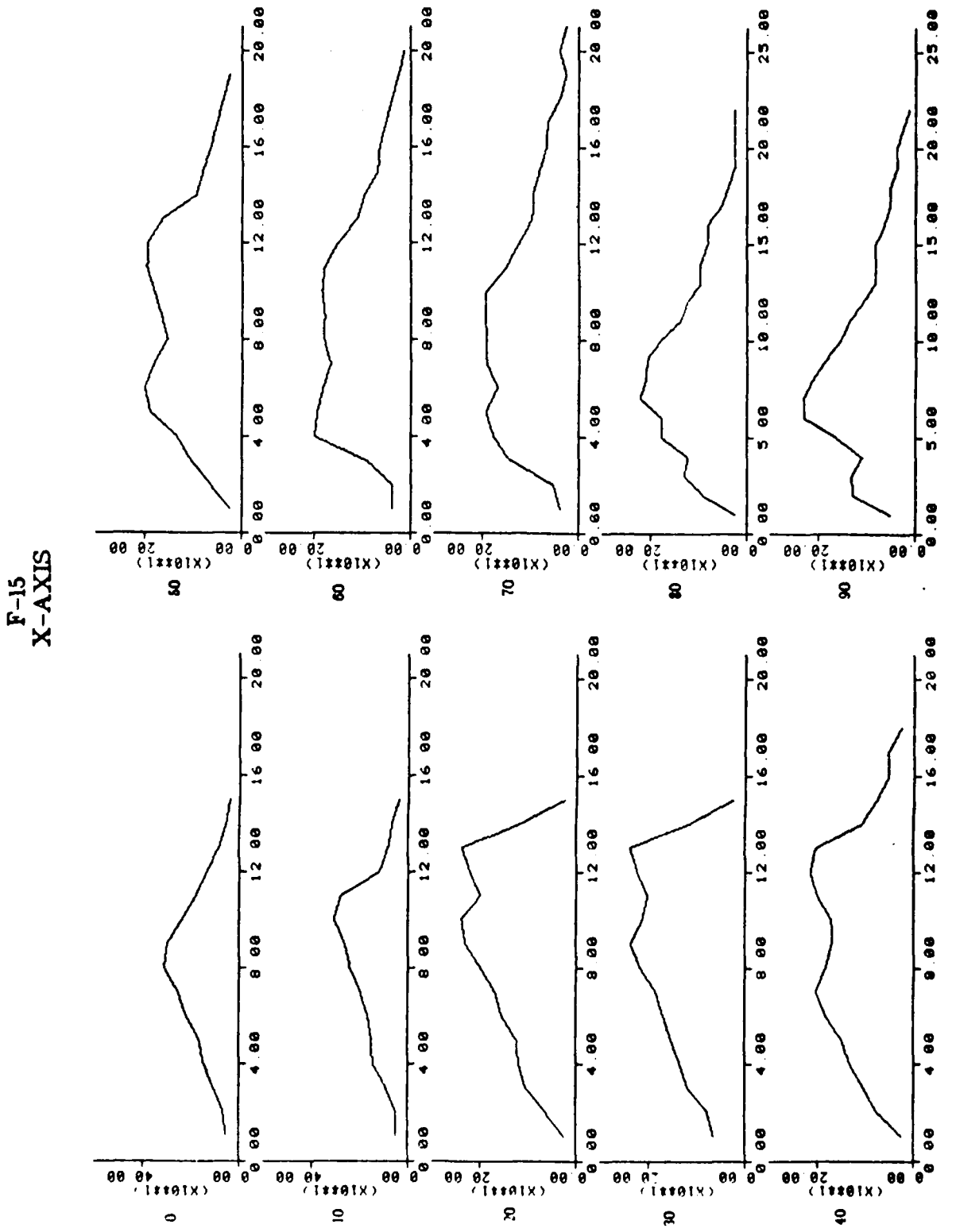


Figure A-7. X-Axis F-15 Data at 10 Degree Increments

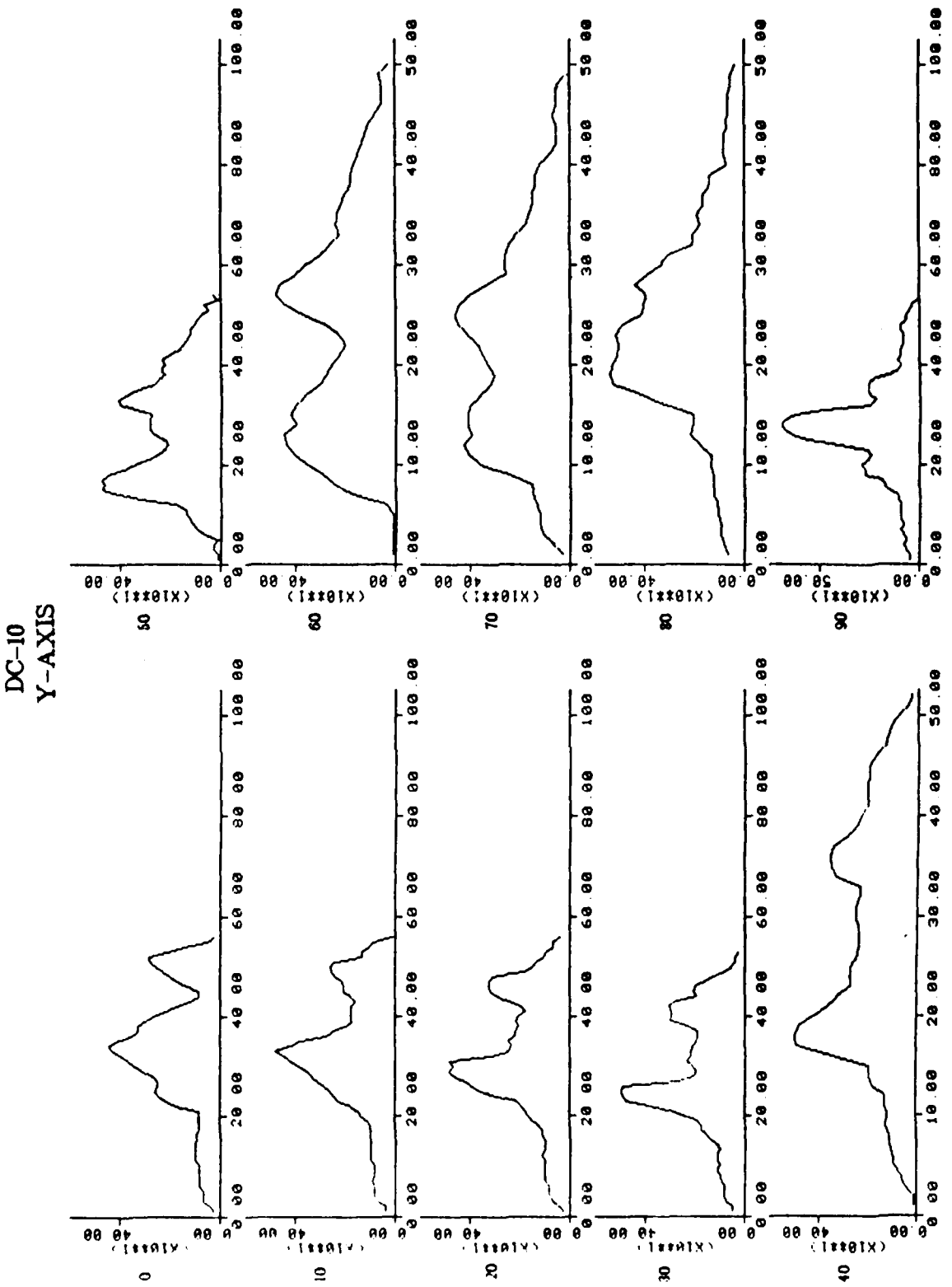


Figure A-6. Y-Axis DC-10 Data at 10 Degree Increments

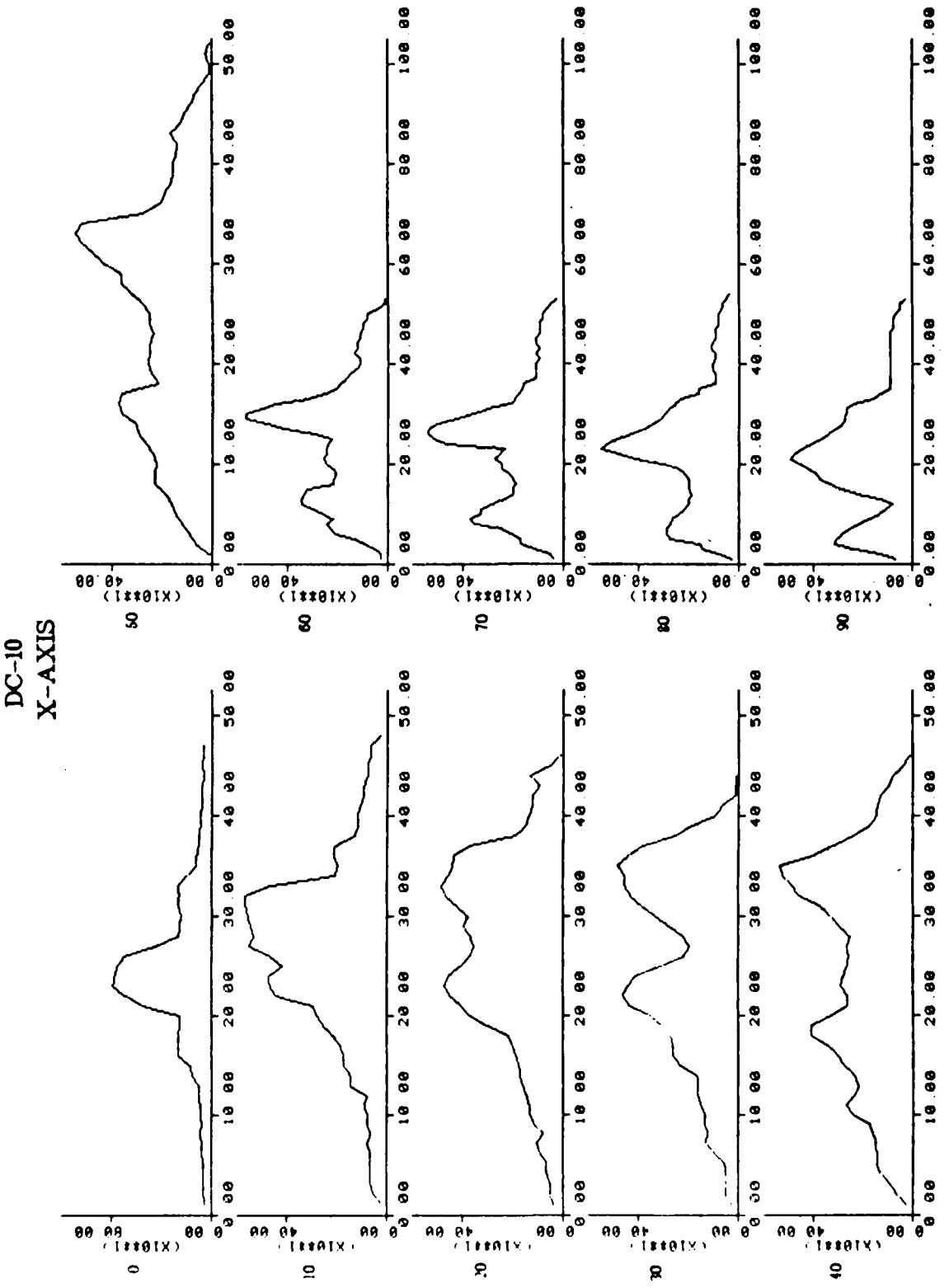


Figure A-5. X-Axis DC-10 Data at 10 Degree Increments

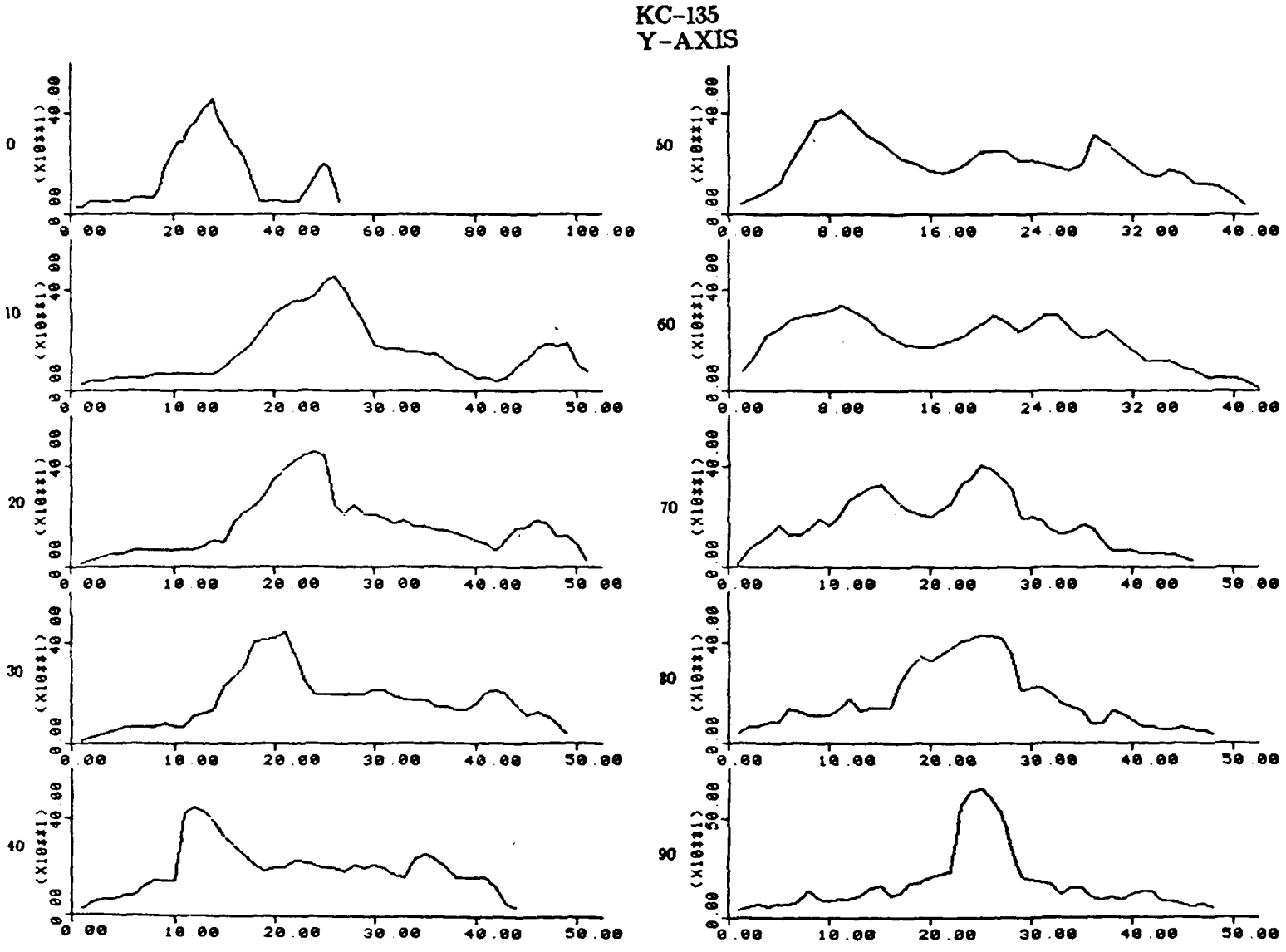


Figure A-4. Y-Axis KC-135 Data at 10 Degree Increments

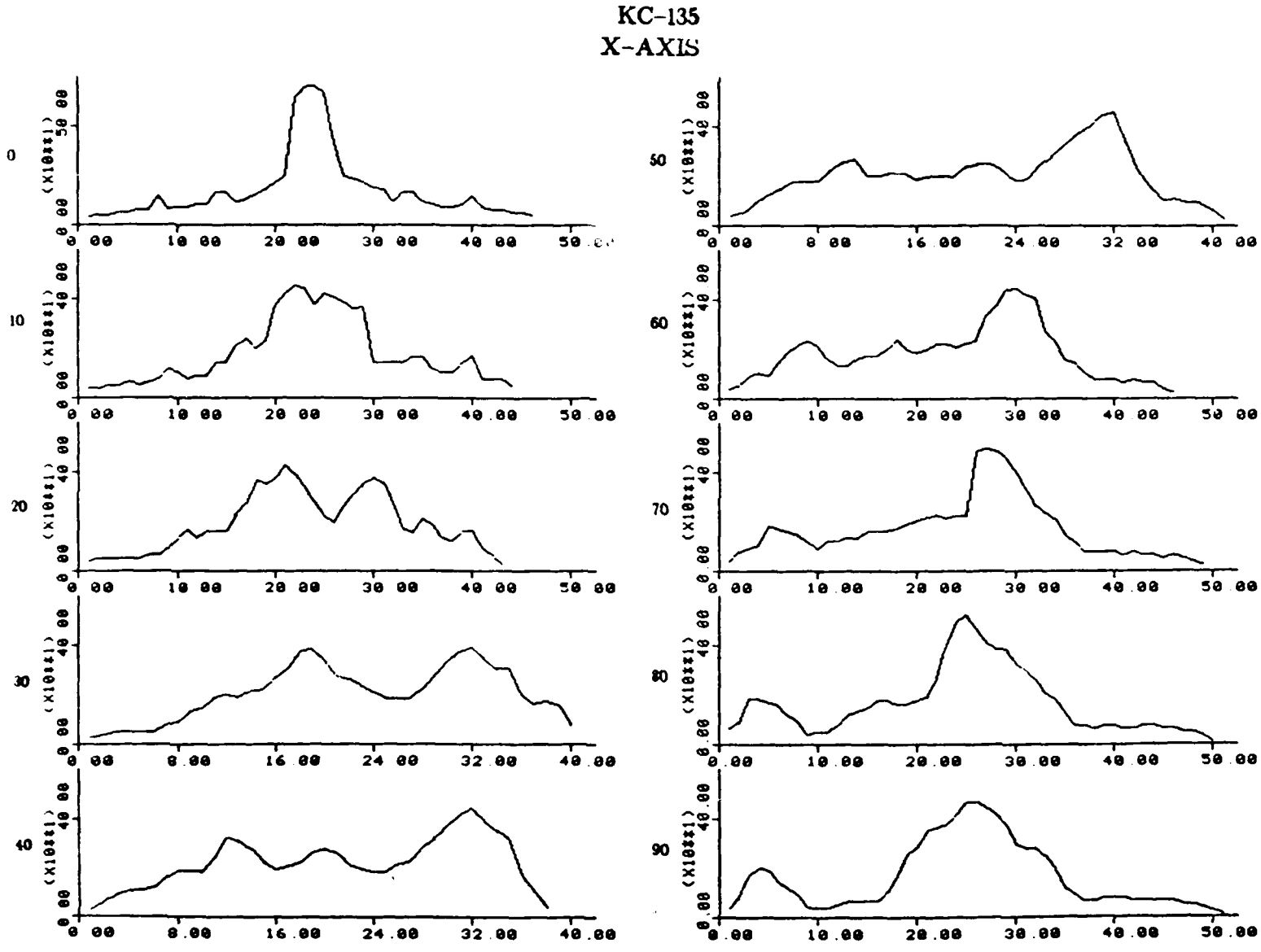


Figure A-3. X-Axis KC-135 Data at 10 Degree Increments

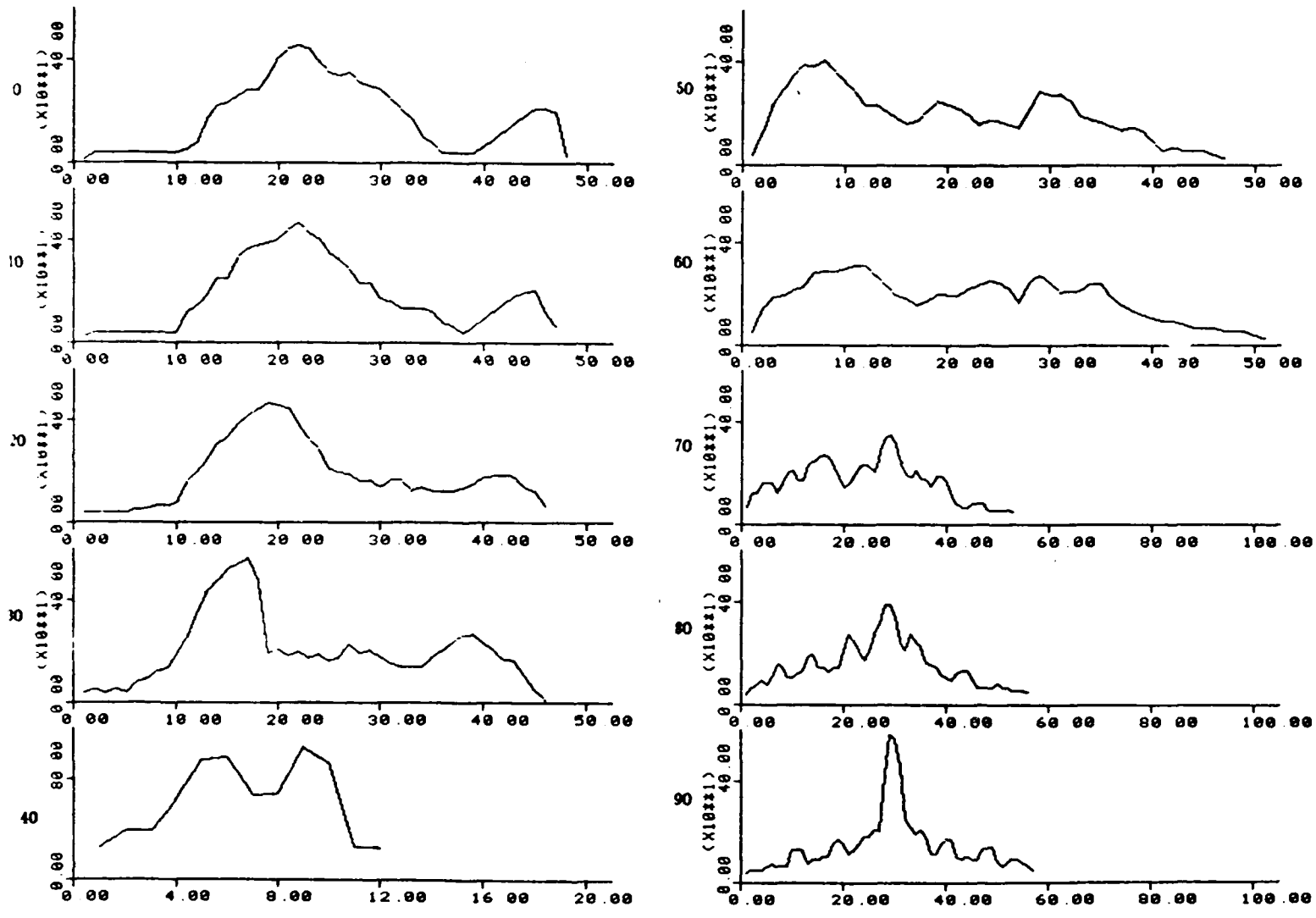


Figure A-2. Y-Axis B-52 Data at 10 Degree Increments

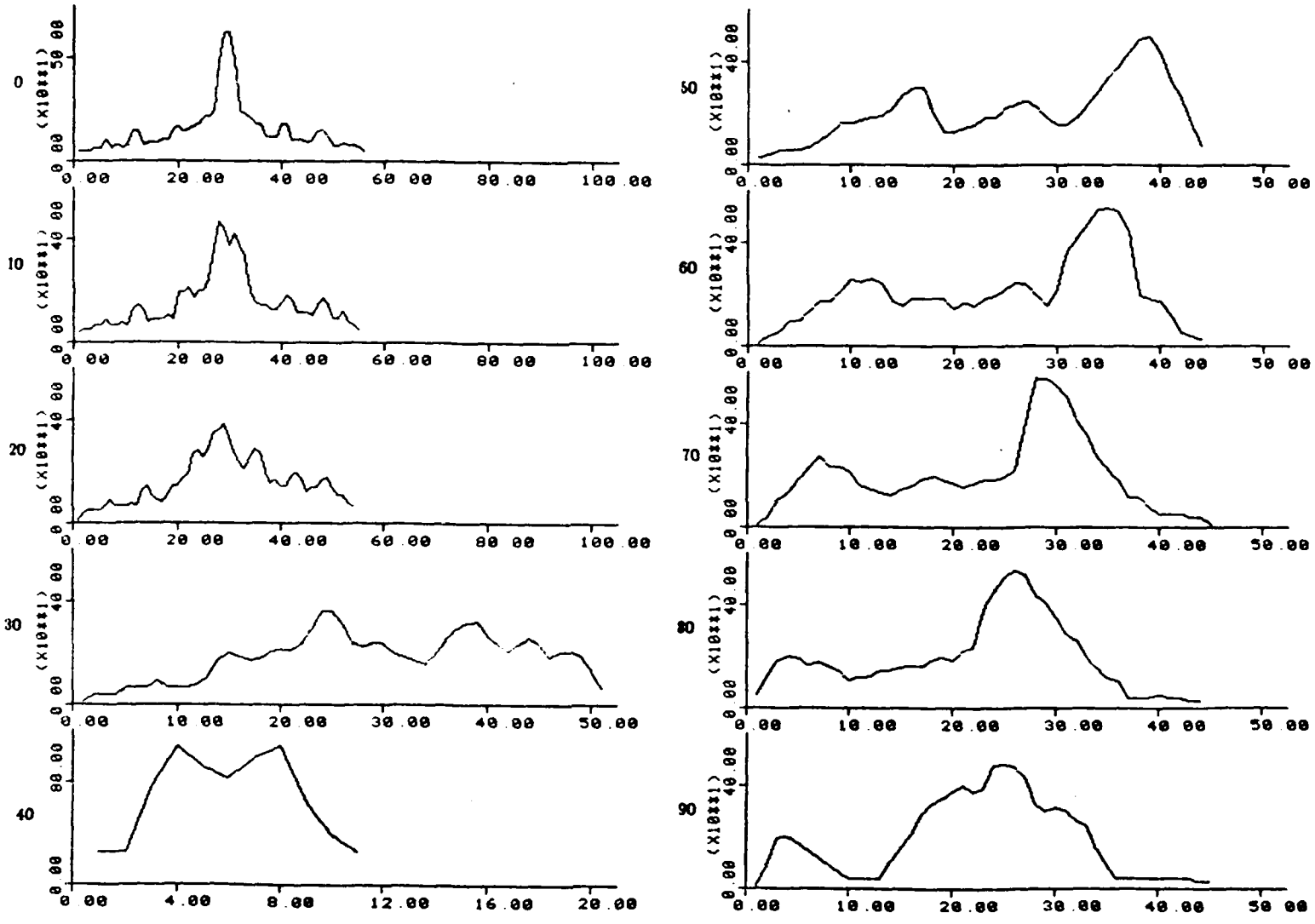


Figure A-1. X-Axis B-52 Data at 10 Degree Increments

```
        CALL OPEN(4,INFILE,1,IER)
        TYPE "READING PROCESSED DATA .."

        READ(4,30) IDIR
30      FORMAT(A2)
        TYPE"DIR: ",IDIR
        READ(4) MPS
        TYPE"MPS: ",MPS
        READ(4) PRESX
        TYPE "X RES: ",PRESX
        READ(4) PRESY
        TYPE "Y RES: ",PRESY
        READ(4) IPX
        IF (ITEST.NE.0) TYPE"IPX: ",IPX
        DO 40 I=1,IPX
                READ(4) XPROC(I)
                IF (ITEST.NE.0) TYPE "XPROC(",I,")= ",XPROC(I)
                IF (XPROC(I).GT.XPMAX) XPMAX=XPROC(I)
                XPINDEX(I)=I
40      CONTINUE

        READ(4) IPY
        IF (ITEST.NE.0) TYPE"IPY: ",IPY
        DO 50 J=1,IPY
                READ(4) YPROC(J)
                IF (ITEST.NE.0) TYPE "YPROC(",J,")= ",YPROC(J)
                IF (YPROC(J).GT.YPMAX) YPMAX=YPROC(J)
                YPINDEX(J)=J
50      CONTINUE

        CALL CLOSE(4,IER)

C
C       COMPARE MODEL AND PROCESSED DATA
C

        DO 400 IP=1,NUMPLANES

                RXMAX=0
                RYMAX=0
                MODFILE(1)="MD"
                MODFILE(2)=IDIR

C       INITIALIZE PROCESSED DATA TEST ARRAYS

                DO 60 I=1,IPX
                        XPTEST(I)=XPROC(I)
                        XPIN(I)=XPINDEX(I)
60              CONTINUE
                DO 70 J=1,IPY
                        YPTEST(J)=YPROC(J)
                        YPIN(J)=YPINDEX(J)
```

```
70              CONTINUE

C       PICK PLANE MODEL

                GO TO (110,120,130,140,150),IP

110             MODFILE(3)="B5"
                MODFILE(4)="2."
                MODFILE(5)="DB"
                MODFILE(6)=0
                GOTO 200
120             MODFILE(3)="KC"
                MODFILE(4)="13"
                MODFILE(5)="5."
                MODFILE(6)="DB"
                MODFILE(7)=0
                GOTO 200
130             MODFILE(3)="DC"
                MODFILE(4)="10"
                MODFILE(5)="X."
                MODFILE(6)="DB"
                MODFILE(7)=0
                GOTO 200
140             MODFILE(3)="F1"
                MODFILE(4)="5."
                MODFILE(5)="DB"
                MODFILE(6)=0
                GOTO 200
150             MODFILE(3)="MI"
                MODFILE(4)="G2"
                MODFILE(5)="1."
                MODFILE(6)="DB"
                MODFILE(7)=0

200     CALL OPEN(3,MODFILE,1,IER)
        WRITE(10,201) (MODFILE(I),I=1,4)
201     FORMAT(1X,"JUST OPENED FILE: ",7A2)
C       GET MODEL DATA

        READ(3) RESOLU
        TYPE"MODEL RESOLUTION: ",RESOLU
        READ(3) IMX
        IF (ITEST.NE.0) TYPE"IMX: ",IMX
        DO 220 I=1,IMX
                READ(3) XMODEL(I)
                IF (ITEST.NE.0) TYPE"XMODEL(",I,")= ",XMODEL(I)
                IF (XMODEL(I).GT.RXMAX) RXMAX=XMODEL(I)
                XMINDEX(I)=I
220     CONTINUE

        READ(3) IMY
        IF (ITEST.NE.0) TYPE"IMY: ",IMY
```

```
        DO 240 J=1,IMY
                READ(3) YMODEL(J)
                IF (YMODEL(J).GT.RYMAX) RYMAX=YMODEL(J)
                YMINDEX(J)=J
240     CONTINUE

        CALL CLOSE(3,IER)

C       NORMALIZE MODEL DATA

        IF (ITEST.NE.0) TYPE"NORMALIZING DATA.."
        IF (ITEST.NE.0) TYPE"RXMAX: ",RXMAX," RYMAX: ",RYMAX
        DO 250 I=1,IMX
                XMODEL(I)=XMODEL(I)/RXMAX
250     CONTINUE
        DO 260 J=1,IMY
                YMODEL(J)=YMODEL(J)/RYMAX
260     CONTINUE

C       NORMALIZE PROCESSED DATA

        IF (ITEST.NE.0) TYPE"XPMAX=",XPMAX," YPMAX=",YPMAX

        XSCALE=FLOAT(IMX)/FLOAT(IPX)
        YSCALE=FLOAT(IMY)/FLOAT(IPY)
        IF (ITEST.NE.0) TYPE"XSCALE: ",XSCALE," YSCALE:",YSCALE

        DO 270 I=1,IPX
                XPTEST(I)=XPTEST(I)/XPMAX
                XPIN(I)=XPIN(I)*XSCALE
270     CONTINUE
        DO 272 J=1,IPY
                YPTEST(J)=YPTEST(J)/YPMAX
                YPIN(J)=YPIN(J)*YSCALE
272     CONTINUE

        IF ((IPR.NE.1).OR.(IP.NE.1)) GOTO  280
                WRITE(12,900)
                WRITE(12,901) (INFILE(L),L=1,10)
                WRITE(12,902)
                WRITE(12,273)
                WRITE(12,274) (XPTEST(L),L=1,IPX)
                WRITE(12,275)
                WRITE(12,274) (YPTEST(L),L=1,IPY)
273             FORMAT(1X,"DUMP OF X PROCESSED DATA")
274             FORMAT(10F6.2)
275             FORMAT(1X,"DUMP OF Y PROCESSED DATA")
                WRITE(12,902)
280     IF (ITEST.EQ.0) GOTO 285
        TYPE"DUMP X OF PROCESSED ARRAYS: "
        WRITE(10,281) (XPTEST(K),K=1,IPX)
281     FORMAT(10F6.2)
```

```
      WRITE(10,281) (XPIN(K),K=1,IPX)
      TYPE
      TYPE"DUMP Y OF PROCESSED ARRAYS:"
      WRITE(10,281) (YPTEST(K),K=1,IPY)
      WRITE(10,281) (YPIN(K),K=1,IPY)
      TYPE
285   IF (IPR.EQ.1) WRITE(12,902)
      IF (IPR.EQ.1) WRITE(12,902)
      IF (IPR.EQ.1) WRITE(12,926)
      IF (IPR.EQ.1) WRITE(12,928) (MODFILE(L),L=1,7)
      IF (IPR.EQ.1) WRITE(12,908) RESOLU,RESOLU
      IF (IPR.EQ.1) WRITE(12,910) IMX,IMY
      IF (IPR.EQ.1) WRITE(12,950)

      CALL INTERP(IPX,XPTEST,XPIN,IMX,XMODEL,XMINDEX,RDIV,ITEST,IPR)
      RXDIV=RDIV
      IF (IPR.EQ.1) WRITE(12,952)
      CALL INTERP(IPY,YPTEST,YPIN,IMY,YMODEL,YMINDEX,RDIV,ITEST,IPR)
      RYDIV=RDIV

      DO 290 I=1,IPX
            XPTEST(I)=XPROC(I)/XPMAX
290   CONTINUE
      DO 295 J=1,IPY
            YPTEST(J)=YPROC(J)/YPMAX
295   CONTINUE

      IF (IPR.EQ.1) WRITE(12,954)
      CALL INTEGRATE(IMX,IPX,RESOLU,PRESX,XMODEL,XPTEST,RDEV,
     *ITEST,IPR)
      RIXDEV=RDEV
      IF (IPR.EQ.1) WRITE(12,956)
      CALL INTEGRATE(IMY,IPY,RESOLU,PRESY,YMODEL,YPTEST,RDEV,
     *ITEST,IPR)
      RIYDEV=RDEV

      CALL LINTEG(IMX,IPX,RESOLU,PRESX,XMODEL,XPTEST,RDEV,RDEV2,
     *ITEST,IPR)
      RI2XDEV=RDEV
      RLXDEV=RDEV2
      CALL LINTEG(IMY,IPY,RESOLU,PRESY,YMODEL,YPTEST,RDEV,RDEV2,
     *ITEST,IPR)
      RI2YDEV=RDEV
      RLYDEV=RDEV2

      RTOTAL=(RESOLU/PRESX)*RXDIV+(RESOLU/PRESY)*RYDIV
      RITOTAL=RIXDEV+RIYDEV
      RI2TOTAL=RI2XDEV+RI2YDEV
      RLTOTAL=RLXDEV+RLYDEV

      GO TO (310,320,330,340,350),IP
```

```
310     BX=RXDIV
        BY=RYDIV
        BT=RTOTAL
        BIX=RIXDEV
        BIY=RIYDEV
        BIT=RITOTAL
        BI2X=RI2XDEV
        BI2Y=RI2YDEV
        BI2T=RI2TOTAL
        BLX=RLXDEV
        BLY=RLYDEV
        BLT=RLTOTAL
        GOTO 400
320     RKX=RXDIV
        RKY=RYDIV
        RKT=RTOTAL
        RIKX=RIXDEV
        RIKY=RIYDEV
        RIKT=RITOTAL
        RI2X=RI2XDEV
        RI2Y=RI2YDEV
        RI2T=RI2TOTAL
        RLX=RLXDEV
        RLY=RLYDEV
        RLT=RLTOTAL
        GOTO 400
330     DX=RXDIV
        DY=RYDIV
        DT=RTOTAL
        DIX=RIXDEV
        DIY=RIYDEV
        DIT=RITOTAL
        DI2X=RI2XDEV
        DI2Y=RI2YDEV
        DI2T=RI2TOTAL
        DLX=RLXDEV
        DLY=RLYDEV
        DLT=RLTOTAL
        GOTO 400
340     FX=RXDIV
        FY=RYDIV
        FT=RTOTAL
        FIX=RIXDEV
        FIY=RIYDEV
        FIT=RITOTAL
        FI2X=RI2XDEV
        FI2Y=RI2YDEV
        FI2T=RI2TOTAL
        FLX=RLXDEV
        FLY=RLYDEV
        FLT=RLTOTAL
        GOTO 400
```

```
350     RMX=RXDIV
        RMY=RYDIV
        RMT=RTOTAL
        RIMX=RIXDEV
        RIMY=RIYDEV
        RIMT=RITOTAL
        RI2MX=RI2XDEV
        RI2MY=RI2YDEV
        RI2MT=RI2TOTAL
        RLMX=RLXDEV
        RLMY=RLYDEV
        RLMT=RLTOTAL

400     CONTINUE

        TYPE
        WRITE(10,500)
        WRITE(10,902)
        WRITE(10,510)
        WRITE(10,902)
        WRITE(10,520) BX,BY,BT,BIX,BIY,BIT
        WRITE(10,522) RKX,RKY,RKT,RIKX,RIKY,RIKT
        WRITE(10,524) DX,DY,DT,DIX,DIY,DIT
        WRITE(10,526) FX,FY,FT,FIX,FIY,FIT
        WRITE(10,528) RMX,RMY,RMT,RIMX,RIMY,RIMT
        WRITE(10,902)
        WRITE(10,530)
        WRITE(10,902)
        WRITE(10,520) BI2X,BI2Y,BI2T,BLX,BLY,BLT
        WRITE(10,522) RI2X,RI2Y,RI2T,RLX,RLY,RLT
        WRITE(10,524) DI2X,DI2Y,DI2T,DLX,DLY,DLT
        WRITE(10,526) FI2X,FI2Y,FI2T,FLX,FLY,FLT
        WRITE(10,528) RI2MX,RI2MY,RI2MT,RLMX,RLMY,RLMT
        TYPE
500     FORMAT(1X,"    SUMMARY OF CORRELATION")
510     FORMAT(10X,"POLY INTERP",30X,"INTEGRATION")
520     FORMAT(1X,"B52:    ",3F10.4,5X,3F10.4)
522     FORMAT(1X,"KC135: ",3F10.4,5X,3F10.4)
524     FORMAT(1X,"DC10:   ",3F10.4,5X,3F10.4)
526     FORMAT(1X,"F15:    ",3F10.4,5X,3F10.4)
528     FORMAT(1X,"MIG21   ",3F10.4,5X,3F10.4)
530     FORMAT(10X,"2ND INTEGRATION",15X,"LINEAR INTERP")

        CALL CLOSE(3,IER)

        IF (IPR.NE.1) GOTO 999
C
C       PRINT RESULTS
C
        DO 800 I=1,24
               WRITE(12,902)
800     CONTINUE
```

```
      WRITE(12,902)
      WRITE(12,902)
      WRITE(12,900)
      WRITE(12,901) (INFILE(I),I=1,10)
      WRITE(12,902)
      WRITE(12,904) IDIR
      WRITE(12,906) MPS
      WRITE(12,908) PRESX,PRESY
      WRITE(12,910) IPX,IPY
      WRITE(12,902)
      WRITE(12,902)
      WRITE(12,912)
      WRITE(12,902)
      WRITE(12,914) BX,BY,BT
      WRITE(12,916) RKX,RKY,RKT
      WRITE(12,918) DX,DY,DT
      WRITE(12,920) FX,FY,FT
      WRITE(12,922) RMX,RMY,RMT
      WRITE(12,902)
      WRITE(12,902)
      WRITE(12,924)
      WRITE(12,902)
      WRITE(12,914) BIX,BIY,BIT
      WRITE(12,916) RIKX,RIKY,RIKT
      WRITE(12,918) DIX,DIY,DIT
      WRITE(12,920) FIX,FIY,FIT
      WRITE(12,922) RIMX,RIMY,RIMT
      WRITE(12,902)
      WRITE(12,902)
      WRITE(12,958)
      WRITE(12,902)
      WRITE(12,914) BI2X,BI2Y,BI2T
      WRITE(12,916) RI2X,RI2Y,RI2T
      WRITE(12,918) DI2X,DI2Y,DI2T
      WRITE(12,920) FI2X,FI2Y,FI2T
      WRITE(12,922) RI2MX,RI2MY,RI2MT
      WRITE(12,902)
      WRITE(12,902)
      WRITE(12,960)
      WRITE(12,902)
      WRITE(12,914) BLX,BLY,BLT
      WRITE(12,916) RLX,RLY,RLT
      WRITE(12,918) DLX,DLY,DLT
      WRITE(12,920) FLX,FLY,FLT
      WRITE(12,922) RLMX,RLMY,RLMT

900   FORMAT(1X,"CORRELATION DATA FOR FILE: ")
901   FORMAT(1X,10A2)
902   FORMAT("                                        ")
904   FORMAT(1X,"DIRECTION:          ",A2,"    DEGREES")
906   FORMAT(1X,"SPEED:              ",F6.2," METERS/S")
908   FORMAT(1X,"TARGET RESOLUTION:  X:",F6.2," Y:",F6.2)
```

```
910     FORMAT(1X,"DATA POINTS:        X:",I3,"   Y:",I3)
912     FORMAT(1X,"DEVIATION SUMMARY (POLYNOMIAL INTERPOLATION)")
914     FORMAT(1X,"     B52:    ",3F8.4)
916     FORMAT(1X,"     KC135:  ",3F8.4)
918     FORMAT(1X,"     DC10:   ",3F8.4)
920     FORMAT(1X,"     F15:    ",3F8.4)
922     FORMAT(1X,"     MIG21:  ",3F8.4)
924     FORMAT(1X,"DEVIATION SUMMARY (INTEGRATION METHOD)")
926     FORMAT(1X,"MODEL FILE: ")
928     FORMAT(1X,7A2)
950     FORMAT(1X,"DATA FOR INTERPOLATION  X-AXIS:" )
952     FORMAT(1X,"DATA FOR INTERPOLATION  Y-AXIS:")
954     FORMAT(1X,"DATA FOR INTEGRATION    X-AXIS:")
956     FORMAT(1X,"DATA FOR INTEGRATION    Y-AXIS:")
958     FORMAT(1X,"DEVIATION SUMMARY (2ND INTEGRATION METHOD)")
960     FORMAT(1X,"DEVIATION SUMMARY (LINEAR INTERPOLATION)")



999     END



C*************************************************************************
C
C       SUBROUTINE DISPLAY DATA
C
C       CALLED BY: PROFILE.FR
C
C       FUNCTION:  DISPLAYS LOW-RESOLUTION SENSOR DATA ON THE
C                  OCTEK IMAGE DISPLAY
C
C       LNAGUAGE:  FORTRAN IV
C
C*************************************************************************

        SUBROUTINE DDATA(IAS,ISUBPIX,IFPX,IFPY,RLOW,ICT)

        INTEGER XPOS,YPOS,IFPX,IFPY,ICT(120)
        REAL RLOW(IAS,IAS),MAX

        IX=38
        IY=20
        ISCALE=ISUBPIX
        MAX=0
        IF (ISCALE.LT.5) ISCALE=5
        IF (ISCALE.GT.10) ISCALE=10

        DO 100 J=1,IFPY
               DO 100 I=1,IFPX
                      IF (RLOW(I,J).GT.MAX) MAX=RLOW(I,J)
100     CONTINUE
```

```
        DO 200 J=1,IFPY
                DO 200 I=1,IFPX

                        RRES=(RLOW(I,J)*15/MAX)
                        IGRAY=IFIX(RRES)
                        XPOS=IX+(I-1)*ISCALE
                        YPOS=IY+(J-1)*ISCALE
                        CALL PXFILL(ICT,IGRAY,XPOS,ISCALE,YPOS,ISCALE)
200     CONTINUE

        RETURN

        END


C*****************************************************************************
C
C       SUBROUTINE DECONVOLUTION
C
C       CALLED BY: PROFILE.FR
C
C       FUNCTION:  DECONVOLVES INPUT TARGET VECTOR WITH RECTANGULAR
C                  OR GAUSSIAN WINDOW AND LOW PASS FILTER'S VECTOR
C                  IN THE FOURIER DOMAIN
C
C       LANGUAGE:  FORTRAN V
C
C*****************************************************************************

        SUBROUTINE DECON(XREAL,RREAL,IWIN,IPTS,ITEST,IPR,BLUR)
        REAL XREAL(64),XIMAG(64),CONV(20),COEFF(10,10)
        REAL RREAL(64),RIMAG(64),WREAL(64),WIMAG(64)

C       CLEAR ARRAYSZ
        DO 10 I=1,64
                XIMAG(I)=0
                RREAL(I)=0
                RIMAG(I)=0
                WREAL(I)=0
                WIMAG(I)=0
                IF (I.GT.15) GOTO 10
                CONV(I)=0
10      CONTINUE

        INUM=32
        ITEMP=IWIN+IPTS
        IF (ITEMP.GT.ITEMP) INUM=64
        NFPNTS=INUM/4+1
        ISG=-1

        DO 100 I=1,INUM
```

```
                RREAL(I)=XREAL(I)
                RIMAG(I)=XIMAG(I)
100     CONTINUE

        TYPE'CALCULATING CONVOLVED TARGET TRANSFORM...'

        CALL FOUREA(XREAL,XIMAG,INUM,ISG)

        IF (ITEST.NE.1) GOTO 290
C       WRITE(10,200)
C       WRITE(10,202) (XREAL(I),I=1,INUM)
C       WRITE(10,208)
C       WRITE(10,202) (XIMAG(I),I=1,INUM)
C       TYPE
        IF (IPR.NE.1) GOTO 290
C       WRITE(12,216)
C       WRITE(12,205) (RREAL(I),I=1,INUM)
C       WRITE(12,217)
C       WRITE(12,205) (RIMAG(I),I=1,INUM)
C       WRITE(12,218)
C       WRITE(12,219)
C       WRITE(12,200)
C       WRITE(12,205) (XREAL(I),I=1,INUM)
C       WRITE(12,208)
C       WRITE(12,205) (XIMAG(I),I=1,INUM)
200     FORMAT(1X,'REAL VALUES:')
202      FORMAT(1X,7F10.4)
205     FORMAT(1X,6F11.4)
C208     FORMAT(1X,'IMAG VALUES:')
C210     FORMAT(1X,'MAGNITUDE:')
C215     FORMAT(1X,'PHASE: ')
C216     FORMAT(1X,"INPUT REAL VALUES: ")
C217     FORMAT(1X,"INPUT IMAG VALUES: ")
218      FORMAT("                              ")
C219     FORMAT(1X,"     CONVOLVED TARGET FFT OUTPUT")
220     FORMAT(1X,"WINDOW TRANSFORM:   (LENGTH=",I3,")")
222     FORMAT(1X,"TARGET- TIME DOMAIN VECTOR")
224      FORMAT(1X,"DECONVOLUTION WINDOW: (LENGTH: ",I3,")")

C       GENERATE DECONVOLUTION WINDOW

290     ILEN=IWIN
        IF (BLUR.EQ.0) GOTO 498
        TYPE"USING GAUSSIAN WINDOW...."

C               GAUSSIAN WINDOW

        RMEAN=IWIN/2+1
        CALL GCOEFF(BLUR,RMEAN,IWIN,COEFF)

        IMEAN=IFIX(RMEAN+.5)
        RMAX=0
```

```
        DO 485 I=1,IWIN
                IF(COEFF(I,IMEAN).LE.RMAX) GOTO 485
                RMAX=COEFF(I,IMEAN)
485     CONTINUE
        DO 490 I=1,IWIN
                CONV(I)=COEFF(I,IMEAN)/RMAX
490     CONTINUE

        IF (IPR.NE.1) GOTO 491
        WRITE(12,492)
        WRITE(12,202) (CONV(I),I=1,IWIN)
492     FORMAT(1X,"GAUSSIAN FUNCTION: ")

CONVOLVE WINDOW

491     ILEN=2*IWIN-1
        DO 495 I=1,ILEN
                DO 495 J=1,IWIN
                        IF ((I-J+1).GT.1WIN) GOTO 495
                        IF ((I-J+1).LT.1) GOTO 495
                        WREAL(I)=WREAL(I)+CONV(I-J+1)
                        IF (IPR.EQ.1) WRITE(12,496) I,WREAL(I)
495     CONTINUE
496     FORMAT(1X,"WREAL(",I2,")= ",F8.3)
        GOTO 500

C               RECTANGULAR WINDOW

498     DO 500 I=1,IWIN
                WREAL(I)=1
500     CONTINUE
        WRITE(10,224) ILEN
        WRITE(10,202) (WREAL(I),I=1,ILEN)
        ISG=-1
        CALL FOUREA(WREAL,WIMAG,INUM,ISG)
        IF (IPR.NE.1) GOTO 525
        WRITE(12,218)
        WRITE(12,220) IWIN
        WRITE(12,200)
        WRITE(12,205) (WREAL(I),I=1,INUM)
C       WRITE(12,208)
C       WRITE(12,205) (WIMAG(I),I=1,INUM)
        WRITE(12,218)

C
C       DIVIDE TRANSFORMS TO DECONVOLUTE TARGET
C

525     IF (NFPNTS.EQ.0) GOTO 550
        ICENTER=INUM/2+1
        ILO=ICENTER-NFPNTS/2
        IHI=ICENTER+NFPNTS/2
```

```
      IF (IPR.EQ.1) WRITE(12,545) NFPNTS,ICENTER,IHI,ILO
545   FORMAT(1X,"FILTERING: ",I3," PNTS, CENTER: ",I3," HI: ",
     *I3," LO: ",I3)
550   DO 600 I=1,INUM
            RDIV=(WREAL(I)*WREAL(I)+WIMAG(I)*WIMAG(I))
            IF (ABS(RDIV).GT.0.00001) GOTO 575
                  RREAL(I)=0
                  RIMAG(I)=0
                  GOTO 585
575         RREAL(I)=(XREAL(I)*WREAL(I)+XIMAG(I)*WIMAG(I))/RDIV
            RIMAG(I)=(WREAL(I)*XIMAG(I)-XREAL(I)*WIMAG(I))/RDIV
585         IF ((I.GE.ILO).AND.(I.LE.IHI)) RREAL(I)=0
            IF ((I.GE.ILO).AND.(I.LE.IHI)) RIMAG(I)=0

600   CONTINUE

C     CHANGE BACK TO TIME DOMAIN

      IF (ITEST.NE.1) GOTO 611
C     TYPE"TARGET DECONVOLUTED F-DOMAIN"
C     TYPE"REAL:"
C     WRITE(10,202) (RREAL(I),I=1,INUM)
C     TYPE"IMAG:"
C     WRITE(10,202) (RIMAG(I),I=1,INUM)
C     TYPE
      IF (IPR.NE.1) GOTO 611
      WRITE(12,218)
      WRITE(12,205) (RREAL(I),I=1,INUM)
C     WRITE(12,205) (RIMAG(I),I=1,INUM)
      WRITE(12,218)
611   ISG=1
      CALL FOUREA(RREAL,RIMAG,INUM,ISG)

C     IF (ITEST.NE.1) GOTO 999
C     TYPE
C     WRITE(10,222)
C     WRITE(10,200)
C     WRITE(10,202) (RREAL(I),I=1,INUM)
C     WRITE(10,208)
C     WRITE(10,202) (RIMAG(I),I=1,INUM)
      IF (IPR.NE.1) GOTO 999
      WRITE(12,218)
      WRITE(12,222)
      WRITE(12,200)
      WRITE(12,205) (RREAL(I),I=1,INUM)
C     WRITE(12,208)
C     WRITE(12,205) (RIMAG(I),I=1,INUM)

999   RETURN
      END
```

```
C*********************************************************************
C
C       PROGRAM: FORT4TO5 - Utility              LANGUAGE: FORTRAN IV
C
C       FUNCTION: CONVERTS TARGET FILES CREATED BY FORTRAN IV PROGRAM
C                 TO FORTRAN V FORMAT COMPATABLE WITH THE ECLIPSE
C                 VERSION OF SCENE.FR
C
C*********************************************************************

        DIMENSION INFILE(13),OUTFILE(13)

        TYPE"ENTER INPUT FORTRAN 4 DATA FILE: "
        READ(11,10) INFILE(1)
10      FORMAT(S13)
        TYPE"ENTER OUTPUT FORTRAN FILENAME:   "
        READ(11,10) OUTFILE(1)

        CALL OPEN(1,INFILE,1,IER)
        IF (IER.NE.1) TYPE"ERROR OPENING INPUT DATA FILE"
        IF (IER.NE.1) STOP

        CALL OPEN(2,OUTFILE,3,IER)
        IF (IER.NE.1) TYPE"ERROR OPENING OUTPUT DATA FILE"
        IF (IER.NE.1) STOP
        READ(1) NPTS
        READ(1) FDIR
        READ(1) TRES
        READ(1) IXL
        READ(1) IYL
        TYPE"NUMBER OF PLANE DATA POINTS: ",NPTS
        TYPE"DIRECTION TARGET IS FACING : ",FDIR
        TYPE"RESOLUTION OF TARGET PIXEL : ",TRES
        TYPE"TARGET DEFINED IN WINDOX X : ",IXL," Y: ",IYL
        WRITE(2,101) NPTS
        WRITE(2,102) FDIR
        WRITE(2,102) TRES
        WRITE(2,101) IXL
        WRITE(2,101) IYL
        DO 100 I=1,NPTS
                READ(1) ITARX
                READ(1) ITARY
                READ(1) ITARI
                WRITE(2,101) ITARX
                WRITE(2,101) ITARY
                WRITE(2,101) ITARI
                TYPE"WRITING POINT: ",I
100     CONTINUE

101     FORMAT(1X,I5)
102     FORMAT(1X,F12.6)
        END
```

```
      SUBROUTINE FOUREA(XREAL,XIMAG,N,ISN)
      DIMENSION XREAL(N),XIMAG(N)

C     FOUREA - VER. 1.0 - 15 JUL 84

C     COMPUTES 1-D FFT USING RADIX 2 DECIMATION IN TIME ALGORITHM
C     DATA AND RESULTS ARE PASSED IN REAL ARRAYS XREAL AND XIMAG
C     ADAPTED BY D. KING FROM IEEE PROGRAMS FOR DIGITAL SIGNAL
C     PROCESSING, SECTION 1.1 (P 1.1-5), BY C.M. RADER.
C     MODIFICATIONS INCLUDE CLEANUP OF CODE, CHECKING OF INPUT
C     PARAMETERS, AND USE OF REAL RATHER THAN COMPLEX ARITHMETIC.

C     CHECK ISN (1=INVERSE FFT, -1=FORWARD FFT)

      ISI=-1
      IF (ISN.EQ.1) ISI=1

C     CHECK FOR POWER OF 2 UP TO 15

      IF (N.LT.2) GO TO 991
      NN=1
      DO 10 I=1,15
      NN=NN*2
      IF (NN.EQ.N) GO TO 20
10    CONTINUE
      GO TO 992
20    CONTINUE

      PI=4.*ATAN(1.)
      PI=FLOAT(ISI)*PI
      FN=N

C     THIS SECTION BIT-REVERSES THE INPUT DATA

      J=1
      DO 80 I=1,N
      IF (I-J) 30,40,40
30    TEMPR=XREAL(J)
      TEMPI=XIMAG(J)
      XREAL(J)=XREAL(I)
      XIMAG(J)=XIMAG(I)
      XREAL(I)=TEMPR
      XIMAG(I)=TEMPI
40    M=N/2
50    IF (J-M) 70,70,60
60    J=J-M
      M=(M+1)/2
      GO TO 50
70    J=J+M
80    CONTINUE

C     COMPUTE BUTTERFLIES
```

```
C*************************************************************************
C
C
C          SUBPROGRAM LINTEG                              LANGUAGE: FORTRAN V
C
C          CALLED BY: COMPARE.FR
C
C          FUNCTION:  COMPARES EXTRACTED TARGET DATA WITH MODEL DATA.
C                     PROGRAM FIRST INTEGRATES MODEL DATA TO MATCH
C                     EXTRACTED DATA, THEN LINEAR INTERPOLATES TO
C                     CONPUTE DEVIATION.
C
C          NOTE: MAIN PROGRAM FOR TARGET VECTOR CLASSIFICATION
C
C*************************************************************************

SUBROUTINE LINTEG(IMPTS,IPPTS,RMRES,RPRES,MODEL,PROC,RDEV,RTOTAL,IT,IP)

        INTEGER IMPTS     ;NUMBER OF MODEL POINTS
        INTEGER IPPTS     ;NUMBER OF POINTS IN TEST CASE

        REAL RMRES,RPRES,MODEL(IMPTS),PROC(IPPTS)
        REAL RDEV,RDEV2,MTEST(100),PTEST(75)
        INTEGER PNXTPNT,PCURPNT,MCURPNT

        IF (IPPTS.GT.3) GOTO 5
                RDEV=0
                RTOTAL=0
                GOTO 999

C       GET RESOLUTION WIDTH

5       INTWD=IFIX(RPRES/RMRES+.5)
        IF (IT.EQ.3) TYPE"INTEGRATION WINDOW: ",INTWD

C       INTEGRATE MODEL TO MATCH TEST CASE

        RWAG=0
        RDEV=0
        ICNT=1
10      CONTINUE
                RPT=0
                DO 20 K=1,INTWD
                        IND = ((ICNT-1)*INTWD+K)
                        IF (IND.GT.IMPTS) GOTO 20
                        RPT=RPT+MODEL(IND)
                        IF (IT.EQ.3) TYPE"ADDING MODEL POINT: ",IND
20              CONTINUE
                IF (IT.EQ.3) TYPE"MODEL POINT(",ICNT,")= ",RPT
                IF (RPT.GT.RWAG) RWAG=RPT
                MTEST(ICNT)=RPT
                ICNT=ICNT+1
```

```
60              OPEN 2,INFILE,ERR=70
                WRITE(10,65) (INFILE(J),J=1,6)
65              FORMAT(1X,"OPENING FILE: ",6A2)
                GOTO 90

C       IF ERROR OPENING FILE THEN HERE

70              WRITE(10,75) (INFILE(J),J=1,6)
75              FORMAT(1X,"ERROR OPENING FILE: ",6A2)
                GOTO 200

C       GET FILE HEADER INFORMATION

90              READ(2,81) RESOLU
                TYPE"FILE RESOLUTION: ",RESOLU
                READ(2,82) IXPNTS
                TYPE"LENGTH OF X-AXIS: ",IXPNTS
                DO 95 J=1,IXPNTS
                        READ(2,82) XDATA(J)
95              CONTINUE
                READ(2,82) IYPNTS
                TYPE"LENGTH OF Y-AXIS: ",IYPNTS
                DO 100 J=1,IYPNTS
                        READ(2,82) YDATA(J)
100             CONTINUE

81              FORMAT(F12.6)
82              FORMAT(I5)
                IF (IAXIS.EQ.2) GOTO 120

110             CALL PLOT10(XDATA,IXPNTS,I,X0,Y0,RESOLU)
                GOTO 130
120             CALL PLOT10(YDATA,IYPNTS,I,X0,Y0,RESOLU)

130             CONTINUE
                CLOSE 2

200     CONTINUE

999     END
```

```
      GOTO (1,2,3,4,5,6,7,8,9,10),I
1           INFILE(2)="00"
            GOTO 15
2           INFILE(2)="10"
            GOTO 15
3           INFILE(2)="20"
            GOTO 15
4           INFILE(2)="30"
            GOTO 15
5           INFILE(2)="40"
            GOTO 15
6           INFILE(2)="50"
            GOTO 15
7           INFILE(2)="60"
            GOTO 15
8           INFILE(2)="70"
            GOTO 15
9           INFILE(2)="80"
            GOTO 15
10          INFILE(2)="90"

15    INFILE(1)="MD"
      GOTO (20,30,40,50,55),IPLANE

20          INFILE(3)="B5"
            INFILE(4)="2."
            INFILE(5)="DF"
            INFILE(6)=0
            GOTO 60
30          INFILE(3)="KC"
            INFILE(4)="13"
            INFILE(5)="5."
            INFILE(6)="DF"
            INFILE(7)=0
            GOTO 60
40          INFILE(3)="DC"
            INFILE(4)="10"
            INFILE(5)="X."
            INFILE(6)="DF"
            INFILE(7)=0
            GOTO 60
50          INFILE(3)="F1"
            INFILE(4)="5."
            INFILE(5)="DF"
            INFILE(6)=0
            GOTO 60
55          INFILE(3)="MI"
            INFILE(4)="G2"
            INFILE(5)="1."
            INFILE(6)="DF"
            INFILE(7)=0
```

```
        WRITE(12,510)   (RDATA(I),I=1,INUM)

999     RETURN
        END

C*************************************************************************
C
C       PLANE INVENTORY PLOTTER - Utility          LANGUAGE: FORTRAN V
C
C       FUNCTION:  PLOTS TARGET DATABASE AT LINE PRINTER.
C                  DATABASE CREATED BY PROGRAM TGEN.FR.  TARGET DATA
C                  HAS A -.DF EXTENTION.
C
C       MACRO:     FORTRAN INVPLOT
C                  RLDR INVPLOT PLOT10 PLOT5.LB @FLIB@
C
C*************************************************************************

        PARAMETER ASIZE=50

        REAL XDATA(100),YDATA(100)
        DIMENSION INFILE(13)

        COMPILER FREE

C       SELECT DATA BASE

        TYPE
        TYPE"            TARGET DATA BASE PLOTTER"
        TYPE
        TYPE"            1) B-52"
        TYPE"            2) KC-135"
        TYPE"            3) DC-10"
        TYPE"            4) F-15"
        TYPE"            5) MIG-21"
        TYPE
        TYPE"           -1) ABORT  "
        TYPE
        ACCEPT"SELECT ONE -> ",IPLANE

        IF (IPLANE.EQ.-1) GOTO 999

        TYPE
        TYPE"            AXIS TO PLOT:"
        TYPE
        TYPE"            1) X-AXIS "
        TYPE"            2) Y-AXIS  "
        TYPE
        ACCEPT"SELECT AXIS -> ",IAXIS
        TYPE

        DO 200 I=1,10
```

```
9               CONTINUE
                F(NP1MK+1)=FLAST
10      CONTINUE

C       CALCULATE PN(X) FOR VARIOUS VALUES OF X

20      RDIV=0
        RLDEV=0
        IVAR=IFIX(FLOAT(INUM)/4.0)
        IBNDRY=INUM-IVAR
        ILBND=IVAR+1
        DO 30 J=ILBND,IBNDRY

                PNOFX=0
                DO 29 I=1, NP1
C                       IF (IT.EQ.1) TYPE'F(',I,')=',F(I),' Y(I)=',Y(I)
                        PNOFX=F(I)+(J-Y(I))*PNOFX
29              CONTINUE

                IF (IT.EQ.1) WRITE(10,600) J,PNOFX
600             FORMAT('NUMBER: ',I4,' F(X): ',F10.2)
                IF (IT.EQ.1) TYPE"MODEL DATA(",J,")= ",RDATA(J)


                GUESS(J)=PNOFX

C       CHECK TO SEE IF MID POINT (J-1) IS RADICAL
C       IF YES, SUBTRACT ITS DEVIATION FROM TOTAL
C
                ILOW=J-2
                IF (ILOW.LT.1) ILOW=1
                IMID=J-1
                IF (IMID.LT.1) IMID=1
                AVG=(GUESS(J)-GUESS(ILOW))/2
                IF (ABS(GUESS(IMID)-AVG).LT.0.8) GOTO 610
                    RDEV=RDEV-RLDEV
                IF (IT.EQ.1) TYPE"DELETING RDEV OF POINT: ",IMID

610             IF ((PNOFX.GT.1).OR.(PNOFX.LT.0)) GOTO 30
                RTEMP=ABS(PNOFX-RDATA(J))*ABS(PNOFX-RDATA(J))
                RDIV=RDIV+RTEMP
                RLDIV=RTEMP
30      CONTINUE
        IF (IT.EQ.1) TYPE'DEVIATION FOR PROCESSED RDATA: ',RDIV
        IF (IPR.NE.1) GOTO 999

        WRITE(12,505)
505     FORMAT(1X,"INTERPOLATED DATA")
        WRITE(12,510) (GUESS(I),I=1,INUM)
510     FORMAT(10F6.2)
        WRITE(12,515)
515     FORMAT(1X,"MODEL DATA")
```

```
303     FORMAT(1X,"INTEGRATION DATA (MODEL): ")
304     FORMAT(1X,"INTEGRATION DATA (TEST):")


305     FORMAT(10F6.2)

999     RETURN
        END

C*****************************************************************************
C
C SUBROUTINE INTERP: MODIFIED FROM 'ELEMENTART NUMERICAL ANALYSIS',BY
C                    S.D CONTE AND CARL DE BOOR (1972), PG 227.
C
C       CALCULATION OF THE NEWTON FORM OF THE POLYNOMIAL OF DEGREE
C       LESS THAN N, WHICH INTERPOLATES AT F(X) AT Y(I), I=1,...,NPI.

        SUBROUTINE INTERP(IMNUM,F,Y,INUM,RDATA,DINDEX,RDIV,IT,IPR)

        INTEGER INUM,NP1,IPNUM
        REAL F(IMNUM),Y(IMNUM),PNOFX,X,DX,RDATA(INUM),DINDEX(INUM)
        REAL GUESS(100),RLAST,RNEXT,RDEV
        X=0
        DX=0

        IF (INUM.GT.3) GOTO 5
                RDEV=0
                GOTO 999

5       NP1=IMNUM
C       CALCULATE THE DIVIDED DIFFERENCES

        N=NP1-1
        IF (N.EQ.0) GOTO 20

        DO 10 K=1,N

                FLOATK=K
                NP1MK=NP1-K
                FLAST=F(1)

                DO 9 I=1,NP1MK

                        IPK=I+K
                        DY=Y(IPK)-Y(I)
                        IF (DY.EQ.0) GOTO 8

                        F(I)=(F(I+1)-FLAST)/DY
                        FLAST=F(I+1)
                        GOTO 9

8                       F(I)=F(I+1)/FLOATK
```

```
                GOTO 10

20      IF (ABS(RMAT-FPT).LE.ABS(RRMAT-FPT)) GOTO 30
                ICNT=ICNT-1

C
C       INTEGRATE MODEL DATA AND NORMALIZE
C

30      RPT=0
        DO 40 I=1, ICNT
                RPT=RPT+MODEL(I)
40      CONTINUE
        MTEST(1)=RPT
        RMAX=RPT

        DO 50 I=2,IPPTS
                RPT=0
                DO 45 K=1,INTWD
                        INDEX=ICNT+K+(I-2)*INTWD
                        IF (INDEX.GT.IMPTS) GOTO 45
                                RPT=RPT+MODEL(INDEX)
45              CONTINUE
                IF (RPT.GT.RMAX) RMAX=RPT
                MTEST(I)=RPT
                IF (IT.NE.2) GOTO 50
                TYPE
                TYPE" CALC POINT: ",I
                TYPE"ENERGY:   MODEL:",RPT," PROC:",PROC(I)
                TYPE"RMAX: ",RMAX
50      CONTINUE

C       NORMALIZE MODEL DATA

        DO 55 I=1,IPPTS
                MTEST(I)=MTEST(I)/RMAX
55      CONTINUE

C       CALCULATE DEVIATION

        RDEV=0
        DO 200 I=1,IPPTS
                RDEV=RDEV+ABS(MTEST(I)-PROC(I))
                IF (IT.EQ.2) TYPE"DEVIATION FOR POINT:",I," IS ",RDEV
200     CONTINUE
        IF (IP.NE.1) GOTO 999

        WRITE(12,303)
        WRITE(12,305) (MTEST(I),I=1,IPPTS)
        WRITE(12,304)
        WRITE(12,305) (PROC(I),I=1,IPPTS)
```

```
C*************************************************************************
C
C       SUROUTINE INTEGRATE              VERSION 1.0
C
C       CALLED BY:  COMPARE.FR
C
C       FUNCTION:   COMPARES EXTRACTED TARGET VECTOR WITH DATABASE
C                   VECTOR
C
C       NOTE: THIS PROGRAM INCLUDED FOR COMPARISON PURPOSES ONLY
C             NOT A RELIABLE COMPARISON TECHNIQUE
C
C*************************************************************************

  SUBROUTINE INTEGRATE(IMPTS,IPPTS,RMRES,RPRES,MODEL,PROC,RDEV,IT,IP)

        INTEGER IMPTS,IPPTS
        REAL RMRES,RPRES,MODEL(IMPTS),PROC(IPPTS),RDEV,MTEST(100)

        IF (IPPTS.GT.3) GOTO 2
                RDEV=0
                GOTO 999

C       GET INTEGRATION WIDTH

2       INTWD=IFIX(RPRES/RMRES)
        IF (IT.EQ.2) TYPE"INTEGRATION WINDOW: ",INTWD

C       INTEGRATE MODEL TO WAG AT NORMALIZATION
        RWAG=0
        DO 5 I=1,IPPTS
                RPT=0
                DO 4 K=1,INTWD
                        IND=(I-1)*INTWD+K
                        IF (IND.GT.IMPTS) GOTO 4
                        RPT=RPT+MODEL(IND)
4               CONTINUE
                IF (RPT.GT.RWAG) RWAG=RPT
5       CONTINUE
        IF (IT.EQ.2) TYPE"MAX ELEMENT=",RWAG," (WAG)"
C
C       START AT LOW END AND MATCH ENERGY
C       AS CLOSE AS POSSIBLE TO FIRST POINT OF PROCESS DATA
C
        FPT=PROC(1)
        RMAT=0
        ICNT=0
10      IF (RMAT.GE.FPT) GOTO 20
                RRMAT=RMAT
                ICNT=ICNT+1
                RMAT= RMAT+(MODEL(ICNT)/RWAG)
                IF (IT.EQ.2) TYPE"NEW NORMALIZED MODEL PNT(",ICNT,"):",RMAT
```

```
        REAL PIX, RLOW(SAR,SAR)

        COMPILER FREE

        IERROR=0
        TYPE'READING DATA ....'
        DO 250 I=1, NBLKX
                DO 250 J=1, NBLKY

                        READ (2) PIX
                        RLOW(J,I)=PIX
C                        TYPE 'PIXEL',J,I,'=',PIX
250     CONTINUE
        GOTO 999

999     RETURN
        END

C ***************************************************************************
C
C       SUBROUTINE GET HEADER -(GETHDR)          LANGUAGE: FORTRAN V
C
C       CALLED BY:  PROFILE.FR
C
C       FUNCTION: OPENS TARGET DATA FILE
C                 READS IN FILE DESCRIPTION HEADER
C
C ***************************************************************************

        SUBROUTINE GETHDR(SAMP,DIR,MPS,FRATE,NBLKX,NBLKY,LOWRES)
        INTEGER SAMP,NBLKX ,NBLKY,FRATE
        REAL LOWRES,DIR,MPS
        DIMENSION INFILE(20)
        TYPE'ENTER NAME OF LOWRES TARGET FILE: '
        TYPE
        READ(11,5) INFILE(1)
5       FORMAT(S20)
        CALL OPEN(2,INFILE)
        READ(2) SAMP
        TYPE 'TARGET FILE HAS: ',SAMP,'FRAMES'
        READ (2) DIR
        TYPE 'DIRECTION:       ',DIR,' RAD'
        READ (2) MPS
        TYPE 'VELOCITY:        ',MPS,' M/S'
        READ (2) FRATE
        TYPE 'FRAME RATE:      ',FRATE,' HZ'
        READ (2) NBLKX, NBLKY
        TYPE 'WINDOW SIZE:     ',NBLKX,' BY ',NBLKY
        READ (2) LOWRES
        TYPE 'DATA RESOLUTION: ',LOWRES,' METERS '
        RETURN
        END
```

```
C*****************************************************************************
C
C SUBROUTINEGaussianCOEFFicient generator    LANGUAGE: FORTRAN V
C
C       CALLED BY:  PROFILE.FR, SCENE.FR
C
C       FUNCTION: RETURNS A COEFFICENT ARRAY CONTAINING THE
C                 DISCRETE VALUES OF A 2-D GAUSSIAN SPREAD FUNCTION
C
C       VARIABLES: SIG    - STANDARD DEVATION
C                           (EQUIVILENT TO (DETSIZE)**0.5)
C                  RM     - MEAN
C                  NUM    - WIDTH
C                  COEFF  - RETURN DATA
C
C*****************************************************************************

        SUBROUTINE GCOEFF(SIG,RM,NUM,COEFF)

        REAL COEFF(NUM,NUM)
        REAL SIG,RM
        INTEGER NUM

        DO 200 I=1,NUM
               DO 200 J=1,NUM

                    COEFF(I,J)=EXP(-(I-RM)*(I-RM)/(2*SIG*SIG))
                    RTEMP=EXP(-(J-RM)*(J-RM)/(2*SIG*SIG))
                    COEFF(I,J)=COEFF(I,J)*RTEMP*100
200     CONTINUE

        DO 300 I=1,NUM
               WRITE(10,301) (COEFF(I,J),J=1,NUM)
300     CONTINUE
301     FORMAT(6X,10F8.2)

        RETURN
        END

C*****************************************************************************
C
C       SUBROUTINE GET DATA (GETDA)          LANGUAGE: FORTRAN V
C
C       CALLED BY:  PROFILE.FR
C
C       FUNCTION:   RETURN 1 FRAME OF DATA TO MAIN
C
C*****************************************************************************

        SUBROUTINE GETDA(SAR,NBLKX,NBLKY,RLOW,IERROR)

        INTEGER NBLKX, NBLKY, IERROR
```

```
      TYPE"THE FILE HAS",INUM," FRAMES"
      ACCEPT"HOW MANY FRAMES TO PRINT? ",IN

      WRITE(12,400) (INFILE(I),I=1,5)
      WRITE(12,405)
      WRITE(12,410) INUM
      WRITE(12,430) ISIZE
      WRITE(12,445) ISP
      WRITE(12,425) IFRATE
      WRITE(12,440) FPRINT
      WRITE(12,450) SIG
      WRITE(12,455) RDRIFT
      WRITE(12,405)
      WRITE(12,435) ILXST,ILYST
      WRITE(12,420) RMPS
      WRITE(12,415) RDIR
      WRITE(12,470) TRES
      WRITE(12,460) ITGRAY
      WRITE(12,465) PIXDIST
      WRITE(12,405)
      WRITE(12,405)
400   FORMAT(1X,"     FRAME DUMP FOR FILE: ",5A2)
405   FORMAT(80X)
410   FORMAT(1X,"FILE CONTAINS:      ",I12," FRAMES")
415   FORMAT(1X,"TARGET DIRECTION:   ",F12.6," RADIANS")
420   FORMAT(1X,"TARGET SPEED:       ",F12.6," M/S")
425   FORMAT(1X,"FRAME  RATE:        ",I12," FRAMES/S")
430   FORMAT(1X,"FOCAL PLANE SIZE:   ",I12," SQ DETECTORS")
435   FORMAT(1X,"TARGET POSITION:    ",I4,I4," @ FRAME 1")
440   FORMAT(1X,"FOOTPRINT:          ",F12.6," METERS")
445   FORMAT(1X,"SIZE OF SUBPIXEL:   ",I12," SQ ")
450   FORMAT(1X,"STD DEV OF OTF:     ",F12.6)
455   FORMAT(1X,"DRIFT RATE:         ",F12.6," METERS/SEC")
460   FORMAT(1X,"TARGET THRESHOLD:   ",I12," GRAYLEVEL")
465   FORMAT(1X,"PIXEL VELOCITY:     ",F12.6," PIXELS/SEC")
470   FORMAT(1X,"TARGET RESOLUTION: ",F12.6," METERS")

      DO 100 I=1,IN
            CALL GETDA(IFPX,ISIZE,ISIZE,RLOW,IERROR)

            WRITE(12,200) I
            WRITE(12,205)
            DO 50 J=1,ISIZE
                  WRITE(12,210) (RLOW(K,J),K=1,ISIZE)
50          CONTINUE
            WRITE(12,205)
            WRITE(12,205)
100   CONTINUE
200   FORMAT(1X,"DATA FRAME: ",I3)
205   FORMAT(1X,"                                        ")
210   FORMAT(1X,20F5.1)
      END
```

```
                MFRAME=HFRAME
                GOTO 300
200     IF ((1-HFRACT).EQ.LFRACT) GOTO 250
                MFRAME=LFRAME
                GOTO 300
250     IF (HFRAME.GT.LFRAME) GOTO 275
                MFRAME=HFRAME
                GOTO 300
275     MFRAME=LFRAME

300     MPIX=IFIX(FRACT*MFRAME+.5)
        AFRAC=FLOAT(MPIX)/FLOAT(MFRAME)
        IF (INEG.EQ.1) MPIX=MPIX*(-1)
        ERROR=ABS(AFRAC-FRACT)

        RETURN
        END

C*************************************************************************
C
C
C       UTILITY FRAME DUMP                      LANGUAGE: FORTRAN IV
C
C       FUNCTION:  Dumps Sensor Data Created By PROGRAM SCENE.FR
C
C*************************************************************************


        PARAMETER IFPX=20,IFPY=20

        REAL RLOW(IFPX,IFPY),RDIR,RMPS,FPRINT
        DIMENSION INFILE(13)

        ACCEPT"ENTER SCENE DATA FILENAME -> ",
        READ(11,11) INFILE(1)
11      FORMAT(S13)
        CALL OPEN(2,INFILE,1,IER)
        READ(2) INUM
        READ(2) ISIZE
        READ(2) ISP
        READ(2) IFRATE
        READ(2) SIG
        READ(2) RDRIFT
        READ(2) RLOWRES
        READ(2) ILXST
        READ(2) ILYST
        READ(2) RMPS
        READ(2) RDIR
        READ(2) TRES
        READ(2) ITGRAY
        READ(2) PIXDIST
        TYPE
```

```
C*****************************************************************************
C
C        SUBROUTINE FRAC                                LANGUAGE:  FORTRAN V
C
C         CALLED BY: PROFILE.FR
C
C         FUNCTION:  CONVERTS THE REMAINDER OF THE SAMPLING RATE
C                    (#DETECTORS/FRAME) TO FRACTION.  THE DENOMINATOR
C                    IS USED TO DETERMINE TARGET PHASING
C
C*****************************************************************************

         SUBROUTINE FRAC(FRACT,MFRAME,MPIX,ERROR)
         REAL FRACT,ERROR,FRACNM,REALNM,HFRACT,LFRACT
         REAL NFRAC,NHFRAC
         INTEGER MFRAME,MPIX,INTNUM,MAXNUM
         INTEGER LFRAME,HFRAME

C        INITIALIZE VARIABLES


         INEG=0
         IF (FRACT.GE.0) GOTO 20
                 FRACT=ABS(FRACT)
                 INEG=1
20       HFRACT=0
         LFRACT=1
         MAXNUM=5

         DO 100 I=1,MAXNUM
C        GET FRACTIONAL PART OF NUMBER
                 REALNM=FRACT*I
                 INTNM=IFIX(REALNM)
                 FRACNM=REALNM-INTNM
C        CHECK MINIMUM LOW FRACTION

                 IF (FRACNM.GE.LFRACT) GOTO 120
                         LFRACT=FRACNM
                         LFRAME=I

C        CHECK MINIMUM LOW FRACTION

120              NFRAC=ABS(1-FRACNM)
                 NHFRAC=ABS(1-HFRACT)
                 IF (NFRAC.GE.NHFRAC) GOTO 100
                         HFRACT=FRACNM
                         HFRAME=I
100      CONTINUE

C        DETERMINE SMALLER FRACTION

         IF ((1-HFRACT).GE.LFRACT) GOTO 200
```

```
        MMAX=1
90      IF (MMAX-N) 100,130,130
100     ISTEP=2*MMAX
        DTHETA=PI/FLOAT(MMAX)
        DO 120 M=1,MMAX
        THETA=DTHETA*FLOAT(M-1)
        C=COS(THETA)
        S=SIN(THETA)
        DO 110 I=M,N,ISTEP
        J=I+MMAX
        TEMPR=C*XREAL(J)-S*XIMAG(J)
        TEMPI=C*XIMAG(J)+S*XREAL(J)
        XREAL(J)=XREAL(I)-TEMPR
        XIMAG(J)=XIMAG(I)-TEMPI
        XREAL(I)=XREAL(I)+TEMPR
        XIMAG(I)=XIMAG(I)+TEMPI
110     CONTINUE
120     CONTINUE
        MMAX=ISTEP
        GO TO 90
130     IF (ISI) 160,140,140

C       FOR INVERSE TRANSFORM (ISI=1) - MULT BY 1/N

140     DO 150 I=1,N
        XREAL(I)=XREAL(I)/FN
        XIMAG(I)=XIMAG(I)/FN
150     CONTINUE

160     RETURN

C       ERRORS

991     TYPE "<7>N = ",N
        STOP FOUREA01 - N MUST BE GREATER THAN 1
992     TYPE "<7>N = ",N
        STOP FOUREA02 - N MUST BE INTEGER POWER OF 2

        END
```

```
        IF (IND.LT.IMPTS) GOTO 10
        IF (IT.EQ.3) TYPE"MAX MODEL POINT: ",RWAG

        ICNT=ICNT-1
        DO 50 I=1,ICNT
                MTEST(I)=MTEST(I)/RWAG
50      CONTINUE

C       TEST DEVIATION FROM NORMALIZED MODEL

        RDEV=0
        I=1
60      IF ((I.GT.ICNT).AND.(I.GT.IPPTS)) GOTO 100
                IF (I.GT.ICNT)  GOTO 65
                   RMPNT=MTEST(I)
                   GOTO 70
65                 RMPNT=0

70              IF (I.GT.IPPTS) GOTO 75
                   RPROC=PROC(I)
                   GOTO 80
75                 RPROC=0
80              RDEV=RDEV+ABS(RMPNT-RPROC)
                IF (IT.EQ.3) TYPE"M PNT: ",RMPNT," P POINT: ",RPROC
                IF (IT.EQ.3) TYPE"DEVIATION OF POINT: ",I," IS ",RDEV
                IF (IT.EQ.3) TYPE
                I=I+1
        GOTO 60
100     IF (IP.NE.1) GOTO 230
        WRITE(12,200) ICNT
        WRITE(12,210) (MTEST(I),I=1,ICNT)
        WRITE(12,220) IPPTS
        WRITE(12,210) (PROC(I),I=1,IPPTS)
200     FORMAT(1X,"NEW INTEGRATION (MODEL): (PNTS= ",I3)
210     FORMAT(1X,8F6.2)
220     FORMAT(1X,"NEW INTEGRATION (TEST CASE): (PNTS= ",I3)

C       LINEAR INTERPOLATE MODEL DATA TO MATCH TEST AND CALC RDEV2

230     RTOTAL=0    ;TOTAL DEVIATION
        PCURPNT=1   ;POINT TO CURRENT TEST POINT
        PNXTPNT=2   ;POINT TO NEXT TEST POINT
        MCURPNT=1   ;POINT TO CURRENT MODEL POINT
        MNXTPNT=2

        RDISTHI=RPRES  ;TEST DATA HIGH POINT (INCREMENTED BY X WIDTH)
        RMMRES=RMRES*INTWD
        IF (IT.EQ.3) TYPE"MODEL RES: ",RMMRES," PROC RES: ",RPRES
240     RTEMP= MCURPNT*RMMRES          ;CHECK INTEGRATION WIN
        IF ((RTEMP.GT.RDISTHI).AND.(MCURPNT.GT.1)) GOTO 350

                IF (IT.EQ.3) TYPE
```

```
                IF (IT.EQ.3) TYPE"CURRENT MODEL POINT: ",MCURPNT

                IF (MCURPNT.LE.ICNT) GOTO 250
                        RMODL=0
                        GOTO 260
250             RMODL=MTEST(MCURPNT)

260             IF (PNXTPNT.LE.IPPTS) GOTO 270  ;CHECK IF END OF DATA
                        RPROC=0
                        GOTO 280

270             RHI=PROC(PNXTPNT)                       ;CALCULATE SLOPE
                RLO=PROC(PCURPNT)
                IF (PCURPNT.LT.1) RLO=0
                RM=(RHI-RLO)/RPRES
                IF (IT.EQ.3) TYPE"SLOPE: ",RM

                B=RHI-RM*RPRES*PNXTPNT                  ;CALC INTERCEPT
                IF (IT.EQ.3) TYPE"INTERCEPT: ",B

                PTEST(MCURPNT)=RM*MCURPNT*RMMRES+B
                RPROC=PTEST(MCURPNT)

280             RDEV2=ABS(RPROC-RMODL)*ABS(RPROC-RMODL)
                RTOTAL=RTOTAL+RDEV2
                IF (IT.NE.3) GOTO 300
                   TYPE"INTERP BETWEEN PROC POINTS: ",PCURPNT,PNXTPNT
                   TYPE"PROC POINT: ",RPROC
                   TYPE"MODEL POINT: ",RMODL
                   TYPE"DEV FOR THIS POINT: ",RDEV2," TOTAL: ",RTOTAL
                   TYPE"RDISTHI: ",RDISTHI
                   TYPE
300             MNXTPNT=MNXTPNT+1
                IF ((XTPNT.GT.IPPTS).AND.(MNXTPNT.GT.ICNT)) GOTO 400
                MCURPNT=MCURPNT+1

        GOTO 240

C       INCREMENT TO NEXT WINDOW

350     PNXTPNT=PNXTPNT+1
        PCURPNT=PCURPNT+1
        RDISTHI=RPRES*PNXTPNT
        GOTO 240

400     IF (IP.NE.1) GOTO 999
        WRITE(12,500) ICNT
        WRITE(12,210) (MTEST(I),I=1,ICNT)
        WRITE(12,510) PCURPNT
        WRITE(12,210) (PTEST(I),I=1,MCURPNT)
500     FORMAT(1X,"LINEAR INTERPOLAT MODEL DATA: (",I3," POINTS)")
510     FORMAT(1X,"LINEAR INTERPOLAT TEST DATA:  (",I3," POINTS:")
```

```
999     RETURN
        END

C*****************************************************************************
C
C                                VERSION 2   26 JULY 1984
C                                VERSION 1
C
C         SUBPROGRAM PIXFRAME                LANGUAGE: FORTRAN V
C
C         FUNCTION:  COMPUTES TARGET VELOCITY IN PIXELS/FRAME
C
C*****************************************************************************

        SUBROUTINE PIXFRAME(DIR,MPS,FRATE,LR,PIXFRX,PIXFRY)

        INTEGER FRATE,SHX,SHY
        REAL MPS,LR,VELX,VELY,PI
        REAL DIR,PIXFRX,PIXFRY

        PI=3.14159265

C       GET COMPONENT VELOCITIES

        VELX=SIN(DIR)*MPS
        VELY=COS(DIR)*MPS

C       INSURE CORRECT SIGN ON VELOCITIES (FOURTH QUAD ORIENTATION)

        IF ((DIR.GT.(PI/2)).AND.(DIR.LT.(1.5*PI))) GOTO 100
                VELX=ABS(VELX)
                GOTO 150
100             VELX=-ABS(VELX)
150     IF ((DIR.GE.0).AND.(DIR.LT. PI)) GOTO 175
                VELY=ABS(VELY)
                GOTO 200
175             VELY=-ABS(VELY)
200     CONTINUE

C       CALCULATE LOWRES PIXEL/FRAME IN X-Y DIRECT

        PIXFRY=VELY/(LR*FRATE)
        PIXFRX=VELX/(LR*FRATE)


        RETURN
        END
```

```
C*************************************************************************
C
C       PROGRAM PIXMOD                          LANGUAGE:  FORTRAN V
C
C       FUNCTION:  USES PICBUF TO CONVERT A STANDARD 256x256x4
C       VIDEO FILE TO 256x256x16 FOR USE BY PROGRAM SCENE.FR
C
C*************************************************************************

        INTEGER IBUF1(300), IBUF2(300)
        INTEGER IARRAY(256)
        DIMENSION INFILE(13), OUTFILE(13)

        TYPE
        TYPE"*OCTEK PICTURE FILE CONVERSION*"
        TYPE
        TYPE"ENTER BACKGROUND VIDEO FILENAME: "
        READ(11,10) INFILE(1)
10      FORMAT(S20)
        TYPE"ENTER OUTPUT FILENAME: "
        READ(11,10) OUTFILE(1)

        TYPE"BUILDING BUFFERS..."
        CALL PICFMT(IBUF1,INFILE,IHDR)
        CALL MAKB(IBUF1)
        CALL PICIN(IBUF1,INFILE,IHDR)

        CALL PFMT(IBUF2,256,256,16,1)
        CALL MAKB(IBUF2)


        DO 100 I=1,256
                TYPE"LINE: ",I
                CALL GROW(IBUF1,I,1,256,IARRAY)
                CALL PROW(IBUF2,I,1,256,IARRAY)
100     CONTINUE

        TYPE"SAVING PICTURE..."
        IHDR=0
        CALL PICOUT(IBUF2,OUTFILE,IHDR)

        CALL RELB(IBUF1)
        CALL RELB(IBUF2)

        END
```

```
C*************************************************************************
C
C       PROGRAM  PROFILE                          LANGUAGE: FORTRAN V
C       FOR THE ECLIPSE
C
C       FUNCTION:  EXTRACTS TARGET SIGNATURE FROM STARING SENSOR
C                  DATA.
C
C       MACRO:     FORTRAN PROFILE
C                 RLDR PROFILE DECON PIXFRAME FRACVPROC FOUREA
C               *  GCOEFF GETHDR GETDA F5PICBUF.LB @FLIB@
C
C*************************************************************************

        PARAMETER MAXWIN=10         ; Number of Sample Windows
        PARAMETER MAXFR=50          ; Max Frames Program Can Process
        PARAMETER LETAR=90          ; Largest Expected Target
        PARAMETER IFPMAX=20         ; Size Of Focal Plane (Detectors)
                                    ; N x N

C       DECLARE ARRAYS

        DIMENSION INFILE(13)
        INTEGER FPLANE(IFPMAX,IFPMAX)
        INTEGER IWINPOSX1(MAXWIN),IWINPOSX2(MAXWIN),IWINPOSY1(MAXWIN)
        INTEGER IWINPOSY2(MAXWIN),XP(MAXWIN),YP(MAXWIN)
        REAL RLOW(IFPMAX,IFPMAX),RXSEQ(MAXWIN,MAXFR)
        REAL XM(MAXWIN),YM(MAXWIN),RYSEQ(MAXWIN,MAXFR)
        REAL XWNOISE(MAXWIN), YWNOISE(MAXWIN)
        REAL XAVG(MAXFR),YAVG(MAXFR),RREAL(MAXFR)

        COMPILER FREE
C
C
C       CLEAR MONITER FOCAL PLANE DISPLAY ARRAY
        DO 5 I=1,IFPMAX
                DO 5 J=1,IFPMAX
                        FPLANE(J,I)="  "
5       CONTINUE

        DO 7 I=1,MAXWIN
                DO 7 J=1,MAXFR
                        RXSEQ(I,J)=0
                        RYSEQ(I,J)=0
                        XAVG(J)=0
                        YAVG(J)=0
                        RREAL(J)=0
7       CONTINUE
        NBND=MAXWIN

C       GET INTERACTIVE DATA
```

```
        IWPRT=0
        ACCEPT"SEND DATA TO PRINTER? ",IPR
        ACCEPT"CHANGE MAX NUMBER OF WINDOWS? :",IANS
        IF (IANS.NE.1) GOTO 8
        ACCEPT"ENTER NUMBER OF WINDOWS: ",NBND
8       ACCEPT"GRAPH WINDOW DATA ?",IANS
        IF (IANS.NE.1) GOTO 10
        TYPE"GRAPH DATA:  1) BEFORE PROCESSING"
        TYPE"             2) AFTER  PROCESSING"
        ACCEPT"             ?",IWPRT

C       GET SCENE HEADER INFORMATION

10      ACCEPT"ENTER SCENE DATA FILE NAME: "
        READ(11,14) INFILE(1)
14      FORMAT(S13)
        OPEN 2,INFILE
        READ(2) INUM     ; NUMBER OF FRAMES
        READ(2) IFPS     ; FOCAL PLANE SIZE
        READ(2) ISP      ; SUB PIXEL RES
        READ(2) IFRATE   ; IFRATE
        READ(2) BLUR     ; OTF STD DEV
        READ(2) DR       ; DRIFT RATE
        READ(2) FPRINT   ; FOOT PRINT
        READ(2) ICTX     ; INITIAL TARG POSITION
        READ(2) ICTY     ;
        READ(2) RMPS
        READ(2) RDIR
        READ(2) TRES     ; PLANE RES
        READ(2) ITH      ; PLANE THRESHOLD
        READ(2) PXDST    ; VEL IN PIX/FRAME
        IFPX=IFPS
        IFPY=IFPS
C
C       PRELIMINARY VELOCITY VECTOR ACQUISITION
C


C
C       DETERMINE PHASE RELATIONSHIP
C          - SETUP WINDOW CONSTRAINTS
C


C       GET TARGET SHIFT

        CALL PIXFRAME(RDIR,RMPS,IFRATE,FPRINT,SHX,SHY)
        TYPE"P/F  X:",SHX," Y:",SHY
        RESX=ABS(SHX*FPRINT)
        RESY=ABS(SHY*FPRINT)

        PIXFRX=ABS(1/SHX)
        PIXFRY=ABS(1/SHY)
```

```
      TYPE"F/P  X:",PIXFRX," Y: ",PIXFRY

      IPIXFRX=IFIX(PIXFRX)
      IPIXFRY=IFIX(PIXFRY)

      FRACX=ABS(PIXFRX-IPIXFRX)
      FRACY=ABS(PIXFRY-IPIXFRY)
      TYPE"FRACX: ",FRACX," FRACY:",FRACY

      IF ((FRACX.GE.0.2).AND.(FRACX.LE.0.8)) GOTO 40
            IWPSX=1
            IWINTX=1
            GOTO 42
40    CALL FRAC(FRACX,IWPSX,IWINTX,ERRORX)
42    IF ((FRACY.GE.0.2).AND.(FRACY.LE.0.8)) GOTO 44
            IWPSY=1
            IWINTY=1
            GOTO 46
44    CALL FRAC(FRACY,IWPSY,IWINTY,ERRORY)
46    TYPE"FOR X DIR: WIN/SET=",IWPSX," NUMERAT:",IWINTX
      TYPE"FOR Y DIR: WIN/SET=",IWPSY," NUMERAT:",IWINTY

      IWPSX=IFIX(IWPSX+.5)
      IWPSY=IFIX(IWPSY+.5)
      NSETX=NBND/IWPSX
      NSETY=NBND/IWPSY
      TYPE"SETS OF WINDOWS:  X: ",NSETX," Y: ",NSETY


C     CALCULATE WINDOW FOR DECONVOLUTION

      IWINCX=IFIX(ABS(1/SHX)*IWPSX+.5)
      IWINCY=IFIX(ABS(1/SHY)*IWPSY+.5)
      TYPE"DECONVOLUTION WINDOW:  X:",IWINCX," Y:",IWINCY
      RESX=(RESX/IWPSX)
      RESY=(RESY/IWPSY)
      TYPE"NEW RES  X: ",RESX," Y: ",RESY
      TYPE
C
C     DETERMINE WINDOW SIZES
C
      IF (RDIR.EQ.0) RDIR=0.01
      IF (RDIR.GT.1.5707) RDIR=1.5707

C     SET INITIAL WINDOW SIZE BY DETERMINING FLITE INTERSECTION

      DX=FLOAT(LETAR)/(SIN(RDIR)*2*FPRINT)
      DY=FLOAT(LETAR)/(COS(RDIR)*2*FPRINT)

      IWINX=INT(2*DX+.5)+2
      IWINY=INT(2*DY+.5)+2
```

```
C       INSURE WINDOW LENGTH ON FOCAL PLANE

        IF (IWINX.LT.1) IWINX=1
        IF (IWINY.LT.1) IWINY=1
        IF (IWINX.GT.IFPX) IWINX=IFPX
        IF (IWINY.GT.IFPY) IWINY=IFPY
        TYPE"WINDOW SIZE    X:",IWINX," Y:",IWINY


C*************************************************************************
C*                                                                       *
C*      DETERMINE X-WINDOW POSITIONS                                     *
C*                                                                       *
C*************************************************************************

C       GET FIRST CENTER

        XDIR=TAN((3.14159265/2)-RDIR)*LETAR/(2*FPRINT)
        IXCENT=ICTX+IFIX(FLOAT(LETAR)/(2*FPRINT)+.5)
        IYCENT=ICTY-IFIX(XDIR)+1

C       INSURE CENTER ON FOCAL PLANE

        IF (IXCENT.LT.1) IXCENT=1
        IF (IYCENT.LT.1) IYCENT=1
        IF (IXCENT.GT.IFPX) IXCENT=IFPX
        IF (IYCENT.GT.IFPY) IYCENT=IFPY

C       STORE FIRST CENTER

        IWINPOSX1(1)=INT(IXCENT+.5)
        IWINPOSX2(1)=INT(IYCENT+.5)

C       GET REMAINING CENTERS RELATIVE TO FIRST CENTER

        RXCENT=FLOAT(IWINPOSX1(1))
        RYCENT=FLOAT(IWINPOSX2(1))
        DO 100 I=2,NBND
75              RXCENT=RXCENT+SIN(RDIR)
                RYCENT=RYCENT-COS(RDIR)
                IX=INT(RXCENT+.5)
                IY=INT(RYCENT+.5)
                IF (IX.GT.IFPX) IX=IFPX
                IF (IY.GT.IFPY) IY=IFPY
                IF (IX.LT.1) IX=1
                IF (IY.LT.1) IY=1
                IF ((IX.EQ.IWINPOSX1(I-1)).AND.(IX.LT.IFPX)) GOTO 75
                        IWINPOSX1(I)=IX
                        IWINPOSX2(I)=IY
100     CONTINUE
        DO 110 I=1,NBND
                TYPE"X WIN COORD: X:",IWINPOSX1(I)," Y:",IWINPOSX2(I)
```

```
110     CONTINUE

C*************************************************************************
C
C       DETERMINE Y-WINDOW POSITIONS
C
C*************************************************************************

C       GET FIRST Y-CENTER

        YDIR=LETAR/(2*FPRINT*TAN((3.14159265/2)-RDIR))
        IXCENT=ICTX+IFIX(YDIR)-1
        IYCENT=ICTY-IFIX(FLOAT(LETAR)/(2*FPRINT)+.5)

C       INSURE CENTER ON FOCAL PLANE

        IF (IXCENT.LT.1) IXCENT=1
        IF (IYCENT.LT.1) IYCENT=1
        IF (IXCENT.GT.IFPX) IXCENT=IFPX
        IF (IYCENT.GT.IFPY) IYCENT=IFPY

C       STORE Y-CENTER

        IWINPOSY1(1)=IXCENT
        IWINPOSY2(1)=INT(IYCENT+.5)

C       GET REMAINING Y-WINDOW CENTERS

        RXCENT=FLOAT(IWINPOSY1(1))
        RYCENT=FLOAT(IWINPOSY2(1))
        DO 200 I=2,NBND
175             RXCENT=RXCENT+SIN(RDIR)
                RYCENT=RYCENT-COS(RDIR)
                IX=INT(RXCENT+.5)
                IY=INT(RYCENT+.5)
                IF (IY.EQ.IWINPOSY2(I-1)) GOTO 175
                        IWINPOSY1(I)=IX
                        IWINPOSY2(I)=IY
200     CONTINUE
        DO 210 I=1,NBND
                TYPE"Y WIN COORD  X:",IWINPOSY1(I)," Y:",IWINPOSY2(I)
210     CONTINUE

C ************************************************************************
C
C       LOOP HERE TO ACQUIRE AND PROCESS DATA
C
C*************************************************************************

        DO 1000 IFR=1,INUM

C       GET FRAME OF DATA AND DISPLAY
```

```
      TYPE'READING DATA ....'
      DO 250 I=1, IFPX
              DO 250 J=1, IFPY
                      READ (2) PIX
                      RLOW(J,I)=PIX
250   CONTINUE

C*********************************************************************
C
C       SUM X-WINDOW DATA
C
C*********************************************************************

              TYPE"SUMMING X-WINDOWS..."
              DO 300 I=1,NBND
                 IXEND=IWINPOSX2(I)+INT(LETAR/(SIN(RDIR)*2*FPRINT))
                 IF (IXEND.GT.IFPY) IXEND=IFPY
                 IXSTART=IXEND-IWINX
                 IF (IXSTART.GT.IFPY) IXSTART=IFPY
                 IF (IXSTART.LT.1) IXSTART=1

                 DO 300 K=IXSTART,IXEND
                    IF (K.LT.1) GOTO 300
                    FPLANE(IWINPOSX1(I),K)="XX"
                    RXSEQ(I,IFR)=RXSEQ(I,IFR)+RLOW(IWINPOSX1(I),K)
                         IF (IFR.NE.1) GOTO 300
                                 XWNOISE(I)=RXSEQ(I,1)

300           CONTINUE

C*********************************************************************
C
C       SUM Y-WINDOW DATA
C
C*********************************************************************

              TYPE"SUMMING Y-WINDOWS..."
              DO 400 I=1,NBND
                 IYSTART=IWINPOSY1(I)-INT(LETAR/(COS(RDIR)*2*FPRINT))
                 IF (IYSTART.LT.1) IYSTART=1
                 IF (IYSTART.GT.IFPX) IYSTART=IFPX
                 IYEND=IYSTART+IWINY
                 IF (IYEND.GT.IFPX) IYEND=IFPX
                 IF (IYEND.LT.1) IYEND=1
                 DO 400 K=IYSTART,IYEND
                    FPLANE(K,IWINPOSY2(I))="YY"
                    RYSEQ(I,IFR)=RYSEQ(I,IFR)+RLOW(K,IWINPOSY2(I))
                    IF (IFR.NE.1) GOTO 400
                        YWNOISE(I)=RYSEQ(I,1)

400                   CONTINUE
```

```
                RTX=FLOAT(ICTX)+(IFR-1)*SHX
                RTY=FLOAT(ICTY)+(IFR-1)*SHY
                IX=INT(RTX+.5)
                IY=INT(RTY+.5)
                IF ((IX.EQ.ICTX).AND.(IY.EQ.ICTY)) GOTO 500
                IF ((IX.GT.IFPX).OR.(IY.GT.IFPY)) GOTO 500
                IF ((IX.LT.1).OR.(IY.LT.1)) GOTO 500
                FPLANE(IX,IY)="TT"
500             CONTINUE

        TYPE"FOCAL PLANE BOUNDRIES .."
        FPLANE(ICTX,ICTY)="**"

        WRITE(10,505) (I,I=1,IFPX)
505     FORMAT(4X,20I3)
        DO 600 I=1,IFPY
                WRITE(10,510) I,(FPLANE(J,I),J=1,IFPY)
510             FORMAT(1X,I3,2X,20(A1,2X))
600     CONTINUE
        TYPE"FRAME: ",IFR

1000    CONTINUE

        IF (IWPRT.NE.1) GOTO 800

        DO 750 I=1,5
                DO 725 J=1,INUM
                        XAVG(J)=RXSEQ(I,J)
725             CONTINUE
                CALL PLOT10(XAVG,INUM,I,X0,Y0,RESX)
750     CONTINUE
        DO 780 I=1,5
                DO 760 J=1,INUM
                        YAVG(J)=RYSEQ(I,J)
760             CONTINUE
                ITEMP=I+5
                CALL PLOT10(YAVG,INUM,ITEMP,X0,Y0,RESY)
780     CONTINUE

C*********************************************************************
C
C       NORMALIZE DATA
C
C*********************************************************************

800     DO 1020 I=1,NBND
                RXMAX=0
                RYMAX=0
                DO 1020 J=1,INUM

                        IF (RXSEQ(I,J).LE.RXMAX) GOTO 1005
```

```
                                RXMAX=RXSEQ(I,J)
1005                    IF (RYSEQ(I,J).LE.RYMAX) GOTO 1015
                                RYMAX=RYSEQ(I,J)
1015                    XM(I)=RXMAX
                        YM(I)=RYMAX
1020    CONTINUE


C               WRITE(10,9000)
C               WRITE(10,9005)
C               DO 1100 I=1,NBND
C                       WRITE(10,9010) (RXSEQ(I,J),J=1,INUM)
C1100           CONTINUE
C               WRITE(10,9005)
C               WRITE(10,9015)
C               WRITE(10,9005)
C               DO 1200 I=1,NBND
C                       WRITE(10,9010) (RYSEQ(I,J),J=1,INUM)
C1200           CONTINUE

9000    FORMAT(1X,"      X WINDOW PROFILE: ")
9005    FORMAT("                                  ")
C9010    FORMAT(1X,20F4.0)
9015    FORMAT(1X,"      Y WINDOW PROFILE: ")
9018    FORMAT(1X,"WINDOW: ",I3)
9020    FORMAT(1X,5F12.6)
9025    FORMAT(1X,"PROFILE TARGET VECTORS FOR FILE: ",5A2)
9027    FORMAT(1X,"FILE HEADER INFORMATION: ")
9030    FORMAT(1X,"FILE CONTAINS:      ",I12," FRAMES")
9031    FORMAT(1X,"FOCAL PLANE SIZE:   ",I12," DETECTORS")
9032    FORMAT(1X,"SUBPIXEL SIZE:      ",I12," SUB PIXELS")
9034    FORMAT(1X,"BLUR DEVIATION:     ",F12.6)
9036    FORMAT(1X,"DRIFT RATE:         ",F12.6," METERS/S")
9037    FORMAT(1X,"PLANE RESOLUTION:   ",F12.6," METERS")
9038    FORMAT(1X,"TARGET THRESHOLD:   ",I12," GRAYLEVEL")
9035    FORMAT(1X,"TARGET DIRECTION:   ",F12.6," RADIANS")
9040    FORMAT(1X,"TARGET VELOCITY:    ",F12.6," METERS/S")
9042    FORMAT(1X,"PIXEL VELOCITY:     ",F12.6," PIX/FRAME")
9045    FORMAT(1X,"FRAME RATE:         ",I12," FRAMES/S")
9050    FORMAT(1X,"FOOTPRINT:          ",F12.6," METERS")
9052    FORMAT(1X,"CONV WINDOW:    X: ",I6," Y: ",I6)
9055    FORMAT(1X,"INITIAL POSITION:   ",I6,I6," DETECTOR(X,Y)")

        IF (IPR.NE.1) GOTO 9250

        WRITE(12,9025) (INFILE(L),L=1,5)
        WRITE(12,9005)
        WRITE(12,9005)
        WRITE(12,9027)
        WRITE(12,9005)
        WRITE(12,9030) INUM
        WRITE(12,9031) IFPS
```

```
      WRITE(12,9032) ISP
      WRITE(12,9045) IFRATE
      WRITE(12,9050) FPRINT
      WRITE(12,9034) BLUR
      WRITE(12,9036) DR
      WRITE(12,9005)
      WRITE(12,9055) ICTX,ICTY
      WRITE(12,9040) RMPS
      WRITE(12,9035) RDIR
      WRITE(12,9037) TRES
      WRITE(12,9038) ITH
      WRITE(12,9042) PXDST
      WRITE(12,9005)
      WRITE(12,505) (I,I=1,IFPX)
      DO 620 I=1,IFPY
            WRITE(12,510) I,(FPLANE(J,I),J=1,IFPX)
620   CONTINUE
      WRITE(12,9052) IWINCX,IWINCY
      WRITE(12,9005)
      WRITE(12,9005)
      WRITE(12,9000)
      WRITE(12,9005)
      DO 9100 I=1,NBND
            WRITE(12,9018) I
            WRITE (12,9020) (RXSEQ(I,J),J=1,INUM)
9100  CONTINUE
      WRITE (12,9005)
      WRITE(12,9015)
      WRITE(12,9005)
      DO 9200 I=1,NBND
            WRITE(12,9018) I
            WRITE(12,9020) (RYSEQ(I,J),J=1,INUM)
9200  CONTINUE

C *****************************************************************
C
C     PROCESS VECTOR PROFILES TO EXTRACT TARGET INFORMATION
C
C *****************************************************************

C     STRIP OFF DC NOISE

9250  DO 9300 I=1,NBND
            IXP=1000
            IYP=1000
            XMAX=0
            YMAX=0
            DO 9300 K=1,INUM
                  RXSEQ(I,K)=RXSEQ(I,K)-XWNOISE(I)
                  IF (RXSEQ(I,K).LE.XMAX) GOTO 9230
                        IF (K.LT.IXP) IXP=K
                        XM(I)=RXSEQ(I,K)
```

```
                                XMAX=XM(I)
9230                        RYSEQ(I,K)=RYSEQ(I,K)-YWNOISE(I)
                            IF (RYSEQ(I,K).LE.YMAX) GOTO 9260
                                IF (K.LT.IYP) IYP=K
                                YM(I)=RYSEQ(I,K)
                                YMAX=YM(I)
9260              XP(I)=IXP
                  YP(I)=IYP

9300      CONTINUE

          IF (IPR.NE.1) GOTO 9450
          WRITE(12,9005)
          WRITE(12,9310)
9310     FORMAT(1X,"     PROFILE WITHOUT DC")
          WRITE(12,9000)
          DO 9400 I=1,NBND
                  WRITE(12,9018) I
                  WRITE(12,9020) (RXSEQ(I,K),K=1,INUM)
9400      CONTINUE
          WRITE(12,9015)
          DO 9450 I=1,NBND
                  WRITE(12,9018) I
                  WRITE(12,9020) (RYSEQ(I,K),K=1,INUM)
9450     CONTINUE

C         NORMALIZE CONVOLUTED VECTORS

C         DO 9500 I=1,NBND
C                 DO 9500 K=1,INUM
C                         RXSEQ(I,K)=RXSEQ(I,K)/XM(I)
C                         RYSEQ(I,K)=RYSEQ(I,K)/YM(I)
C9500     CONTINUE
C
C         IF (IPR.NE.1) GOTO 9550
C         WRITE(12,9005)
C         WRITE(12,9510)
C9510    FORMAT(1X,"      NORMALIZED WITHOUT DC")
C         WRITE(12,9000)
C         DO 9530 I=1,NBND
C                 WRITE(12,9018) I
C                 WRITE(12,9020) (RXSEQ(I,K),K=1,INUM)
C9530    CONTINUE
C         WRITE(12,9015)
C         DO 9550 I=1,NBND
C                  WRITE(12,9018) I
C                  WRITE(12,9020) (RYSEQ(I,K),K=1,INUM)
C9550    CONTINUE

C
C         PROCESS LOW SNR VECTORS
C
```

```
                IF (ICON.NE.1) GOTO 594
                DO 593 I=1,IXLEN
                     IF (CONVX(I).GT.X1MAX) X1MAX=CONVX(I)
593             CONTINUE
                DO 594 I=1,IYLEN
                     IF (CONVY(I).GT.Y1MAX) Y1MAX=CONVY(I)
594             CONTINUE
                DO 595 I=1,IXL
                     RXVECT(I)=FLOAT(XVECT(I))/XMAX
595             CONTINUE
                DO 596 I=1,IYL
                     RYVECT(I)=FLOAT(YVECT(I))/YMAX
596             CONTINUE
                IF (ICON.NE.1) GOTO 598
                DO 597 I=1,IXLEN
                     RCONVX(I)=FLOAT(CON      ))/X1MAX
597             CONTINUE
                DO 598 I=1,IYLEN
                     RCONVY(I)=FLOAT(CONVY(I))/Y1MAX
598             CONTINUE
                WRITE(12,310)
                WRITE(12,350) IXNUM
                WRITE(12,599) (RXVECT(I),I=1,IXL)
                WRITE(12,370) IYNUM
                WRITE(12,599) (RYVECT(I),I=1,IYL)
                IF (ICON.NE.1) GOTO 600
                WRITE(12,310)
                WRITE(12,599) (RCONVX(I),I=1,IXLEN)
                WRITE(12,511)
                WRITE(12,599) (RCONVY(I),I=1,IYLEN)
599             FORMAT(1X,10F8.5)
C
C       GRAPH OUTPUT
C
600     ACCEPT" DO YOU WANT TO GENERATE PLOTS OF OUTPUT? ",IGRPH
        IF (IGRPH.NE.1) GOTO 750
C       ACCEPT"PLOT (1) CONVOLVED  OR  (2) NORMAL DATA? ",ICN
C       ACCEPT"PLOT (1) X AXIS     OR  (2) Y AXIS     ? ",IAXIS
C
C       TYPE"ENTER TITLE OF GRAPH: "
C       READ(11,3) INFILE(1)
C
C       TYPE"FLIP TERMINAL SWITCH...AND HIT ANY KEY..."
C       PAUSE
C       NG=1
C       MD =0
C       YN=0
C       YX=0
C       IFS=0
C       IF ((ICN.NE.1).AND.(IAXIS.NE.1)) GOTO 760
C               CALL GR PH2(INFILE,NG,YVECT,XVECT,IYNUM,MD,YN,YX,IFS)
C60     IF ((ICN.NE.1).AND.(IAXIS.EQ.1)) GOTO 770
```

```
330     FORMAT(1X,"TARGET WINDOW:          ",I4," BY ",I4)
340     FORMAT(1X,"TARGET RESOLUTION:      ",F10.6," METERS")
350     FORMAT(1X,"X-VECTOR:     ",I4," ELEMENTS")
360     FORMAT(20I4)
370     FORMAT(1X,"Y-VECTOR:     ",I4," ELEMENTS")
380     FORMAT(20I4)
C
C       CONVOLVE DATA
C
400     ACCEPT"DO YOU WANT TO CONVOLVE DATA? ",ICON
        IF (ICON.NE.1) GOTO 590

        ACCEPT"ENTER WINDOW SIZE->",IWIN
        IXLEN=IXL+IWIN-1
        IYLEN=IYL+IWIN-1
        DO 500 I=1,IXLEN

                DO 500 J=1,IWIN
                        IF ((I-J+1).LT.1) GOTO 500
                        CONVX(I)=CONVX(I)+XVECT(I-J+1)
500             CONTINUE
        DO 505 I=1,IYLEN
                DO 505 J=1,IWIN
                        IF((I-J+1).LT.1) GOTO 505
                        CONVY(I)=CONVY(I)+YVECT(I-J+1)
505     CONTINUE
        WRITE(10,510)
        WRITE(10,380) (CONVX(I),I=1,IXLEN)
510     FORMAT(1X,"X VECTOR CONVOLUTION")
        WRITE(10,511)
511     FORMAT(1X,"Y VECTOR CONVOLUTION")
        WRITE(10,380) (CONVY(I),I=1,IYLEN)

        IF (IPR.NE.1) GOTO 575
                WRITE(12,510)
                WRITE(12,380) (CONVX(I),I=1,IXLEN)
                WRITE(12,511)
                WRITE(12,380) (CONVY(I),I=1,IYLEN)

590     ACCEPT"PRINT NORMALIZED VECTORS? ",IANS
        IF (IANS.NE.1) GOTO 600

                XMAX=0
                YMAX=0
                X1MAX=0
                Y1MAX=0
                DO 591 I=1,IXL
                      IF (XVECT(I).GT.XMAX) XMAX=XVECT(I)
591             CONTINUE
                DO 592 I=1,IYL
                      IF (YVECT(I).GT.YMAX) YMAX=YVECT(I)
592             CONTINUE
```

```
                DO 67 I=1,IXL
                    WRITE BINARY(3) HIRES(I,J)
67      CONTINUE

55      TYPE"CALCULATING X-VECTOR ..."
        DO 60 J=1,IXL
                DO 50 I=1,NUMPOINTS
                        IF (TARGETX(I).EQ.J) XVECT(J)=XVECT(J)+15-TARGETI(I)
                        IF (TARGETX(I).GT.1) GOTO 50
                TYPE"POSSIBLE NOISE DATA POINT:"
                TYPE"TARGET POINT: ",TARGETX(I),TARGETY(I),"=",TARGETI(I)
50              CONTINUE
                IF (XVECT(J).GT.0) IXNUM=IXNUM+1
60      CONTINUE
        TYPE"NUMBER X-ELEMENTS= ",IXNUM
        TYPE"CALCULATING Y-VECTOR: "
        DO 100 J=1,IYL
                DO 70  I=1,NUMPOINTS
                        IF (TARGETY(I).EQ.J) YVECT(J)=YVECT(J)+15-TARGETI(I)
                        IF (TARGETY(I).GT.1) GOTO 70
                        IF (TARGETY(I).LT.(IYL-1)) GOTO 70
                        TYPE"POSSIBLE NOISE DATA POINT:"
                TYPE"TARGET POINT: ",TARGETX(I),TARGETY(I),"=",TARGETI(I)
70              CONTINUE
                IF (YVECT(J).GT.0) IYNUM=IYNUM+1
100     CONTINUE
        TYPE"NUMBER OF Y-ELEMENTS= ",IYNUM
        IDIR=IFIX(180/3.14159265*FDIR+.01)
        TYPE
        TYPE"DATA FOR TARGET FILE: "
        TYPE
        TYPE"FACING DIRECTION: ",IDIR
        TYPE"WINDOW:           ",IXL," BY ",IYL
        TYPE"RESOLUTION:       ",RESOLU
        TYPE
        TYPE"X-VECTOR: "
        WRITE(10,200) (XVECT(I),I=1,IXL)
        TYPE"Y-VECTOR: "
        WRITE(10,200) (YVECT(I),I=1,IYL)
200     FORMAT(20I4)

        IF (IPR.NE.1) GOTO 400

        WRITE(12,310)
        WRITE(12,350) IXNUM
        WRITE(12,360) (XVECT(I),I=1,IXL)
        WRITE(12,370) IYNUM
        WRITE(12,380) (YVECT(I),I=1,IYL)

300     FORMAT(1X,"VECTOR DATA FOR TARGET MODEL DATA FILE: ",S20)
310     FORMAT(80X)
320     FORMAT(1X,"TARGET DIRECTION:      ",I6," DEGREES")
```

```
C       OUTPUT HEADER

        IF (IPR.NE.1) GOTO 42

        WRITE(12,300) INFILE(1)
        WRITE(12,310)
        WRITE(12,320) IDIR
        WRITE(12,330) IXL,IYL
        WRITE(12,340) RESOLU
        WRITE(12,310)

42      TYPE"CLEARING VECTOR MATRICES..."
        DO 45 I=1,70
                XVECT(I)=0
                CONVY(I)=0
                CONVX(I)=0
                YVECT(I)=0
45      CONTINUE
        ACCEPT"CHANGE GREYSCALE? ",IANS
        IF (IANS.NE.1) GOTO 575
                ACCEPT"ENTER GREYSCALE THRESHOLD->",ITHRES
                DO 47 I=1,NUMPOINTS
                        IF (TARGETI(I).GT.ITHRES) TARGETI(I)=ITHRES
47              CONTINUE

C       SET-UP 2-D ARRAY

575             DO 578 I=1,70
                        DO 578 J=1,70
                                HIRES(I,J)=0
578             CONTINUE
                DO 580 I=1,NUMPOINTS
                        HIRES(TARGETX(I),TARGETY(I))=15-TARGETI(I)
580             CONTINUE

                ACCEPT"PRINT 2- PLOT OF TARGET FILE? ",IANS
                IF (IANS.NE.1) GOTO 65
                WRITE(12,310)
                WRITE(12,584) (INFILE(I),I=1,5),ITHRES
                DO 582 J=1,IYL
                        WRITE(12,585) (HIRES(I,J),I=1,IXL)
582             CONTINUE
584             FORMAT(1X,"HIRES PLOT OF FILE: ",5A2," THRESHOLD: ",I3)
585             FORMAT(1X,34I3)

65      ACCEPT"GENERATE BINARY FILE? ",IANS
        IF (IANS.NE.1) GOTO 55
        ACCEPT"ENTER FILENAME ->"
        READ(11,3) INFILE(1)
        CALL OPEN(3,INFILE,3,IER)
        DO 67 J=1,IYL
```

```
        TYPE
        ACCEPT"DO YOU WANT ANOTHER RUN? ",IANS
        IF (IANS.EQ.1) GOTO 15

        IF (IPR.EQ.1) CLOSE 3
        CALL RELB(IBUF)
        END

C*************************************************************************
C
C
C       TARGET EVALUATOR :TARGEVAL      VERSION 2.0
C
C       FUNCTION: THIS PROGRAM GENERATES X AND Y VECTOR PROFILES
C       FOR TARGET DATA FILES, AND EITHER SAVES DATA:
C               1) LINE PRINTER
C               2) DATABASE
C
        DIMENSION INFILE(20)
        INTEGER NUMPOINTS, TARGETX(900),TARGETY(900),TARGETI(900)
        INTEGER XVECT(70), YVECT(70), CONVX(70),HIRES(70,70)
        INTEGER CONVY(70)
        REAL RXVECT(70),RYVECT(70),RCONVX(70),RCONVY(70)


1       IXNUM=0
        IYNUM=0
        TYPE
        TYPE"ENTER TARGET FILENAME: "
        READ(11,3) INFILE(1)
3       FORMAT(S13)

        ACCEPT"SEND DATA TO PRINTER? ",IPR
        TYPE"OPENING FILE..."
        CALL OPEN(4,INFILE,1,IER)
                IF (IER.NE.1) TYPE "CANT OPEN TARGET FILE: ",IER
                IF (IER.NE.1) STOP

        READ(4) NUMPOINTS
        READ(4) FDIR
        TYPE"# OF POINTS: ",NUMPOINTS," DIRECTION: ",FDIR
        READ(4) RESOLU
        READ(4) IXL
        READ(4) IYL
        TYPE"WINDOW: ",IXL,IYL
        DO 40 I=1,NUMPOINTS
                READ(4) TARGETX(I)
                READ(4) TARGETY(I)
                READ(4) TARGETI(I)
40      CONTINUE
        CALL CLOSE(4,IER)
        TYPE"FINISHED READING DATA..."
```

```
        IYEND=IBKOFFY+WINSIZE-1
        ISCALE=DETSIZE

        DO 200 J=IYSTART,IYEND,ISCALE

C       FILL BUFFER WITH ISCALE ROWS OF DETECTORS
                DO 175 L=1,ISCALE
                        IROW=J+L-1
                        ICOL=IXSTART
                        CALL GROW(IBUF,IROW,ICOL,WINSIZE,IDETBUF(1,L))

175             CONTINUE

C       INTEGRATE ROWS

                IBND=WINSIZE-1
                DO 190 I=1,IBND,ISCALE
                        IPIX=0
                        DO 180 K=1,ISCALE
                                DO 180 L=1,ISCALE
                                        IROW=I+K-1
                                        ICOL=L
                                        IPIX=IPIX+IDETBUF(IROW,ICOL)
180                     CONTINUE
                        RPIX=FLOAT(IPIX)/FLOAT(DETSIZE*DETSIZE)
                        IF (IPR.EQ.1) WRITE(3) RPIX
                        ICTR=I/ISCALE+1
                        RDETROW(ICTR)=RPIX
190             CONTINUE
                WRITE(10,189) (RDETROW(M),M=1,NDET)
189             FORMAT(10X,20F3.0)
200     CONTINUE

        IF (IPAUSE.EQ.1) PAUSE

C       CALCULATE NEW TARGET POSITION

        IF (ITAR.NE.1) GOTO 500
        DELTAX=SIN(RDIR)*PIXDIST
        DELTAY=(-1)*COS(RDIR)*PIXDIST

        CXPOS=CXPOS+DELTAX
        CYPOS=CYPOS+DELTAY

C       DRIFT SCENE

        INSX=IDSX+INSX

        CALL PICIN(IBUF,BKGFILE,IHDR)

500     CONTINUE
```

```
                WRITE(3) RMPS
                WRITE(3) RDIR
                WRITE(3) TRES
                WRITE(3) ITGRAY     ; PLANE THRESHOLD
                WRITE(3) PIXDIST
30      CONTINUE

C
C       GENERATE LOW RESOLUTION SCENE SEQUENCES
C

        DO 500 ICNT=1,INUM

                TYPE"DISPLAYING FRAME: ",ICNT

C       MASK BUFFERED BACKGROUND WITH TARGET

                IF (ITAR.NE.1) GOTO 45

                IXPOS=IFIX(CXPOS)
                IYPOS=IFIX(CYPOS)
                TYPE"CURRENT TARGET POSITION X:",IXPOS," Y:",IYPOS
                DO 45 I=1,NPTS
                        IX=TARGETX(I)+IXPOS
                        IY=TARGETY(I)+IYPOS
                        IV=15-TARGETI(1)
                        IF (IV.LT.1) GOTO 45
                        IF (IV.GT.15) IV=15
                        IF (ITGRAY.LT.0) GOTO 40
                        IF (IV.LT.ITGRAY) IV=ITGRAY
40                      CALL PPNT(IBUF,IY,IX,IV)
45              CONTINUE

C
C       CONVOLVE WITH POINT-SPREAD FUNCTION
C

        IF (NPSF.NE.1) GOTO 192
        TYPE"CONVOLVING PSF ..."
        TYPE"CALCULATING COEFF, M=",RMEAN," SIGMA=",SIG
        CALL GCOEFF(SIG,RMEAN,DETSIZE,COEFF)
        TYPE"CALCULATINE OTF ..."
        CALL PSPREAD(DETSIZE,IBKOFFX,IBKOFFY,INSX,WINSIZE,IBUF,COEFF)

C
C       INTEGRATE HIRES WINDOW AND DISPLAY AS FOCAL PLANE PIXELS
C


192     IXSTART=IBKOFFX+INSX
        IXEND=IBKOFFX+WINSIZE+INSX
        IYSTART=IBKOFFY
```

```
        RDIR=RDIR*0.01745329

        ITGRAY=-1
        ACCEPT"DO YOU WANT TO CHANGE GRAYSCALE OF TARGET? ",IANS
        IF (IANS.NE.1) GOTO 20
                ACCEPT"ENTER GRAYSCALE FOR TARGET (0-15): ",ITGRAY

20      ACCEPT"ENTER STARTING POSITION (X,Y): ",IXST,IYST
        CXPOS=IXST+IBKOFFX
        CYPOS=IYST+IBKOFFY

        ACCEPT"ENTER FRAME RATE: ",IFRATE
        PIXDIST=RMPS/(IFRATE*TRES)
        RLOWRES=DETSIZE*TRES

C
C       ESTABLISH NOISE PARAMETERS
C

25      IDSX=IFIX(RDRIFT/IFRATE+.5)
        IDSX=IFIX(IDSX/TRES+.5)
        RDRIFT=IDSX*TRES*FLOAT(IFRATE)
        TYPE"DRIFT RATE: ",RDRIFT,"METERS/SEC"
        TYPE"PIXELS/FRA: ",IDSX

C       TIE UP LOOSE ENDS
        ACCEPT"ENTER NUMBER OF FRAMES IN THIS SEQUENCE: ",INUM
        ACCEPT"PAUSE AFTER EACH FRAME? ",IPAUSE

C
C       OPEN AND SETUP OUTPUT DATA FILE
C

        ACCEPT"DO YOU WANT TO GENERATE A SCENE DATA FILE? ",IPR
        IF (IPR.NE.1) GOTO 30
                TYPE"ENTER OUTPUT FILENAME: "
                READ(11,13) OUTFILE(1)
                OPEN 3,OUTFILE
                NUMDET=NDET

                ILXST=IXST/DETSIZE+IXL/(2*DETSIZE)+1
                ILYST=IYST/DETSIZE+IYL/(2*DETSIZE)+1
                TYPE"FOCAL PLANE TARGET COORD:  X:",ILXST," Y:",ILYST
                WRITE(3) INUM       ; NUMBER OF FRAMES
                WRITE(3) NUMDET     ; SIZE OF FOCAL PLANE
                WRITE(3) ISIZE      ; SIZE OF SUB PIXELS
                WRITE(3) IFRATE     ; FRAME RATE
                WRITE(3) SIG        ; STD DEV OF OPTICS BLUR
                WRITE(3) RDRIFT     ; DRIFT RATE
                WRITE(3) RLOWRES    ; FOOTPRINT
                WRITE(3) ILXST      ; INITIAL TARGET POSITION
                WRITE(3) ILYST
```

```
        OPEN 2,TARFILE,ERR=101
        GOTO 7
101             TYPE"CAN'T OPEN TARGET FILE, ERROR:",IER
                STOP
7       ACCEPT"SIMULATE SENSOR DRIFT? ",IANS
        IF (IANS.NE.1) GOTO 8
        ACCEPT"ENTER DRIFT RATE IN METERS/SEC: ",RDRIFT

8       ACCEPT"SIMULATE OPTICAL TRANSFER FUNCTION? ",NPSF
        IF (NPSF.NE.1) GOTO 9
                ACCEPT"ENTER SIGMA OF OPTICAL BLUR: ",SIG
                RMEAN=DETSIZE/2+1

C       SET UP MEMORY BUFFERS FOR BACKGROUND VIDEO FILE

9       TYPE"BUILDING BUFFERS..."
        CALL PICFMT(IBUF,BKGFILE,IHDR)
        CALL MAKB(IBUF)
        CALL PICIN(IBUF,BKGFILE,IHDR)

C       READ TARGET DATA FILE

        IF (ITAR.NE.1) GOTO 25
        TYPE
        TYPE"READING TARGET DATA FILE..."
        READ(2,98) NPTS
        READ(2,99) FDIR
        READ(2,99) TRES
        READ(2,98) IXL
        READ(2,98) IYL
        DO 10 I=1,NPTS
                READ(2,98) TARGETX(I)
                READ(2,98) TARGETY(I)
                READ(2,98) TARGETI(I)
10      CONTINUE
98      FORMAT(I5)
99      FORMAT(F12.6)
        CLOSE 2

        TYPE
        WRITE(10,11) (TARFILE(I),I=1,3)
11      FORMAT(1X,"     DATA FOR TARGET FILE: ",3A2)
        TYPE"NUMBER OF DATA POINTS: ",NPTS
        TYPE"TARGET FACING:         ",FDIR," DEGREES"
        TYPE"TARGET RESOLUTION:     ",TRES," METERS "
        TYPE"TARGET SIZE:           ",IXL," BY ",IYL
        TYPE

C       GET TARGET SPECIFICATIONS

15      ACCEPT"ENTER TARGET VELOCITY  (MPS):  ",RMPS
        ACCEPT"ENTER TARGET DIRECTION (DEG):  ",RDIR
```

```
      PARAMETER LORGX=155,LORGY=50
      PARAMETER TX=600,TY=600,TI=600
      PARAMETER IBKOFFX=30,IBKOFFY=30          ; DET=5,  BKG=78
                                               ; DET=9,  BKG=30
      PARAMETER NDET=20                        ; DET=15, BKG=45
      PARAMETER DETSIZE=9                      ; DET=20, BKG=28
      PARAMETER WINSIZE=180

      COMPILER FREE

C     DECLARE VARIABLES
      INTEGER XIN,YIN

C     DECLARE ARRAYS

      DIMENSION BKGFILE(20),TARFILE(20),OUTFILE(20)
      INTEGER IBUF(NBUFSZ)
      INTEGER TARGETX(TX),TARGETY(TY),TARGETI(TI)
      INTEGER IDETBUF(WINSIZE,DETSIZE)
      REAL COEFF(DETSIZE,DETSIZE), RDETROW(WINSIZE)

C     INITIALIZE VARIABLES AND GET DATA FILES

      NUMDET=NDET
      ISIZE=DETSIZE
      IBKGWIN=WINSIZE
      RDRIFT=0
      IDSX=0
      INSX=0
      SIG=0

      TYPE
      TYPE"    SCENE GENERATOR          VERSION 2.5"
      TYPE"                             FOR ECLIPSE"
      TYPE
      TYPE"ENTER BACKGROUND VIDEO FILENAME -> "
      READ(11,13) BKGFILE(1)
13    FORMAT(S20)

      ACCEPT"SIMULATE TARGET? ",ITAR
      IF (ITAR.EQ.1) GOTO 6
              RDIR=0
              RMPS=0
              IFRATE=0
              IXL=0
              IYL=0
              IXST=0
              IYST=0
              RLOWRES=0
              GOTO 7
6     TYPE"ENTER TARGET DATA FILENAME -> "
      READ(11,13) TARFILE(1)
```

```
                IROW=I+BKOFFY-1
                ICOL=BKOFFX+INSX
C               WRITE(12,180) (IDONE(L),L=1,20)
                CALL PROW(IBUF,IROW,ICOL,WINSIZE,IDONE)
C               CALL GROW(IBUF,IROW,ICOL,WINSIZE,IDONE)
C               WRITE(12,180)  (IDONE(L),L=1,20)
300     CONTINUE
        RX=FLOAT(MX)

C
C       NORMALIZE PICTURE TO 4 BIT DISPLAY (0-15)
C
        TYPE"NORMALIZING FOV ..."
        DO 400 J=1,WINSIZE

                IROW=J+BKOFFY-1
                ICOL=BKOFFX+INSX
                CALL GROW(IBUF,IROW,ICOL,WINSIZE,IDONE)
                RATIO=FLOAT(15)/RX
                DO 350 I=1,WINSIZE
                        IDONE(I)=IFIX(FLOAT(IDONE(I))*RATIO+.75)
                        IF (IDONE(I).GT.15) IDONE(I)=15
350             CONTINUE
                CALL PROW(IBUF,IROW,ICOL,WINSIZE,IDONE)
400     CONTINUE

        RETURN
        END

 C*****************************************************************
 C                                                                 C
 C       PROGRAM SCENE                       VERSION 2.5           C
 C       AUGUST 15, 1984                     FOR THE ECLIPSE       C
 C                                                                 C
 C       FUNCTION:  GENERATES SIMULATED FOCAL PLANE OUTPUT         C
 C                  TO TEST RESOLUTION ENHANCEMENT ROUTINES.       C
 C                                                                 C
 C       INPUT:  1) BACKGROUND VIDEO FILE                          C
 C               2) TARGET DESCRIPTION FILE (FROM TGEN.FR)         C
 C               3) DESIRED NOISE STATISTICS                       C
 C               4) DESIRED TARGET CHARACTERISTICS                 C
 C                  (SPEED,DIRECTION,..)                           C
 C                                                                 C
 C       OUTPUT: SEQUENCE OF FOCAL PLANE FRAMES CONTAINING         C
 C               TARGET AND BACKGROUND INFORMATION                 C
 C                                                                 C
 C*****************************************************************C

 C       DECLARE CONSTANTS

        PARAMETER NBUFSZ=300
        PARAMETER HORGX=45, HORGY=50
```

```
40              CONTINUE

        TYPE"WIDTH OF PSF: ",PNTSPD
C       GET FIRST N-1 ROWS TO SET-UP FOR CONVOLUTION

        IBND=PNTSPD-1
        DO 50 I=1,IBND
                IROW=BKOFFY-(PNTSPD/2)+I-1
                ICOL=BKOFFX+INSX-(PNTSPD/2)
                IWIDTH=WINSIZE+PNTSPD-1
                CALL GROW(IBUF,IROW,ICOL,IWIDTH,IARRAY(1,I))
50      CONTINUE

C       SET-UP ARRAY POINTER FOR PNTSPD WIDTH CONVOLUTION

        DO 100 I=1,PNTSPD
                IPTR(I)=I-1
100     CONTINUE

C
C       COMPUTE CONVOLUTION

        MX=0
        DO 300 I=1,WINSIZE

C       COMPUTE NEW POINTERS

                DO 150 K=1,PNTSPD
                        IPTR(K)=MOD(IPTR(K),PNTSPD)+1
150             CONTINUE

C       GET ROW N

                IROW=I+(PNTSPD/2)+BKOFFY-1
                CALL GROW(IBUF,IROW,ICOL,IWIDTH,IARRAY(1,IPTR(PNTSPD)))
                TYPE"STARTING ROW: ",I
                DO 200 J=1,WINSIZE
                        RVAL=0
                        DO 175 K=1,PNTSPD
                                DO 175 L=1,PNTSPD
                                        IX=K+J-1
                                        IY=IPTR(L)
                                        RVAL=RVAL+COEFF(K,L)*FLOAT
     1(IARRAY(IX,IY))
                                        IF (RVAL.GT.32000) RVAL=32000
175                     CONTINUE
                        IDONE(J)=IFIX(RVAL+.5)
C80                    FORMAT(1X,25I5)
                        IF (IDONE(J).GT.MX) MX=IDONE(J)
200             CONTINUE

C       SAVE NEWLY COMPUTED ROW
```

```
        WRITE(3,9756) RESX
        WRITE(3,9756) RESY
        WRITE(3,9757) IXL
        DO 9750 I=1,IXL
                INDEX=IXL-I+1
                WRITE(3,9756) XAVG(INDEX)
9750    CONTINUE
        WRITE(3,9757) IYL
        DO 9755 I=1,IYL
                WRITE(3,9756) YAVG(I)
9755    CONTINUE
9756    FORMAT(1X,F12.6)
9757    FORMAT(1X,I5)

        CLOSE 3
9999    END

C*****************************************************************************
C
C       SUBROUTINE POINT-SPREAD
C
C       FUNCTION: CONVOLVES ANY 2-D PICTURE INDEXED BY PICBUF ARRAY
C                 WITH A COEFFICIENT ARRAY TO SIMULATE A OPTICAL
C                 POINT-SPREAD FUNCTION. (CAN BE USED FOR ANY SIZE
C                 2-D CONVOLUTION).
C
C       VARIABLES: DETSIZE - SIZE OF CONVOLUTION MATRIX
C                  BKOFFX  - OFFSET OF SCENE VIEWING WINDOW IN X DIR
C                  BKOFFY  - OFFSET OF SCENE   "        "    " Y DIR
C                  INSX    - ADDITIONAL OFFSET IN X DIRECTION (DRIFT)
C                  WINSIZE - SIZE OF VIEWING WINDOW (WINSIZE,WINSIZE)
C                  IBUF    - UTILITY ACCESS FOR SCENE (USED BY PICBUF)
C                  COEFF   - CONVOLUTION COEFFICIENTS
C                  MX      - MAXIMUM POINT COMPUTED (FOR AVERAGING)
C
C*****************************************************************************

    SUBROUTINE PSPREAD(DETSIZE,BKOFFX,BKOFFY,INSX,WINSIZE,IBUF,COEFF)


        INTEGER IARRAY(200,9), IDONE(190)
        INTEGER IPTR(9),IBUF(300)
        REAL COEFF(DETSIZE,DETSIZE)

        INTEGER PNTSPD, DETSIZE, BKOFFX,BKOFFY,WINSIZE

C       INSURE POINT-SPREAD RANGE IS ODD

        IF ((DETSIZE/2-FLOAT(DETSIZE)/2).NE.0) GOTO 20
                PNTSPD=DETSIZE-1
                GOTO 40
20              PNTSPD=DETSIZE
```

```
C PREPROCESS X-VECTORS

      MWIN=MAXWIN
      ITEST=1
      IF (IPR.EQ.1) WRITE(12,9000)
      CALL VPROC(MWIN,IWPSX,NSETX,MAXFR,NBND,NAVG,XWNOISE,RXSEQ,XP,IPR,XAVG,I
      CALL DECON(XAVG,RREAL,IWINCX,IXL,ITEST,IPR,BLUR)
      IXL=IXL-IWINCX+1
      DO 9555 I=1,MAXFR
               XAVG(I)=ABS(RREAL(I))
              RREAL(I)=0
9555  CONTINUE
      IF (IWPRT.NE.2) GOTO 9556
      CALL PLOT10(XAVG,INUM,1,X0,Y0,RESX)

9556  IF (IPR.NE.1) GOTO 9963
               WRITE(12,9005)
               WRITE(12,9562)
9562           FORMAT(1X,"      X TARGET VECTOR")
               WRITE(12,9020) (XAVG(I),I=1,IXL)
               WRITE(12,9005)
               WRITE(12,9005)

C PREPROCESS Y-VECTORS

9963  IF (IPR.EQ.1) WRITE(12,9015)
      CALL VPROC(MWIN,IWPSY,NSETY,MAXFR,NBND,NAVG,YWNOISE,RYSEQ,
     *YP,IPR,YAVG,IYL)
      CALL DECON(YAVG,RREAL,IWINCY,IYL,ITEST,IPR,BLUR)
      IYL=IYL-IWINCY+1
      DO 9565 I=1,MAXFR
      YAVG(I)=ABS(RREAL(I))
9565  CONTINUE
      IF (IWPRT.NE.2) GOTO 9566
      CALL PLOT10(YAVG,INUM,10,X0,Y0,RESY)

9566    IF (IPR.NE.1) GOTO 9980
               WRITE(12,9005)
               WRITE(12,9567)
9567           FORMAT(1X,"      Y TARGET VECTOR")
               WRITE(12,9020) (YAVG(I),I=1,IYL)

9980    ACCEPT"SAVE TARGET VECTORS? ",ISVECT
        IF (ISVECT.NE.1) GOTO 9999
        OPEN 3,"COMP.DA"
        ACCEPT"ENTER TARGET DIRECTION: "
        READ(11,9966) IDIR
9966    FORMAT(A2)
        WRITE(3,9968) IDIR
9968    FORMAT(A2)
        WRITE(3,9756) RMPS
```

```
C               CALL GR PH2(INFILE,NG,XVECT,YVECT,IXNUM,MD,YN,YX,IFS)
C70     IF (IAXIS.NE.1) GOTO 780
C               CALL GR PH2(INFILE,NG,CONVY,CONVX,IYLEN,MD,YN,YX,IFS)
C80             CALL GR PH2(INFILE,NG,CONVX,CONVY,IXLEN,MD,YN,YX,IFS)
C
C
C       GENERATE DATABASE
C
750     ACCEPT"DO YOU WANT TO GENERATE A DATA BASE FILE? ",IANS
        IF (IANS.NE.1) GOTO 999

        INFILE(1)="MD"

        TYPE
        TYPE"         CURRENT PLANE INVENTORY:    "
        TYPE
        TYPE"            1 - 6) B-52"
        TYPE"            2 - 7) KC-135"
        TYPE"            3 - 8) DC-10"
        TYPE"            4 - 9) F-15"
        TYPE"            5 - 10) MIG-21"
        TYPE
        TYPE"            1ST NUMBER - MODEL DATABASE FILE"
        TYPE"            2ND NUMBER - COMPARE TEST FILE"
        TYPE
        ACCEPT"            WHICH ONE? ",IPICK

        TYPE"ENTER DIRECTION: (MUST BE '2' DIGIT INTEGER - 00, 09, 45, ETC) "
        READ (11,405) INFILE(2)
405     FORMAT(A2)
        ACCEPT"PROCESS IN FORTRAN FIVE FORMAT? ",IFIVE

        GO TO (410,420,430,440,450,460,470,480,490,495),IPICK

410     INFILE(3)="B5"
        INFILE(4)="2."
        INFILE(5)="DB"
        IF (IFIVE.EQ.1) INFILE(5)="DF"
        INFILE(6)=0
        GOTO 800
420     INFILE(3)="KC"
        INFILE(4)="13"
        INFILE(5)="5."
        INFILE(6)="DB"
        IF (IFIVE.EQ.1) INFILE(6)="DF"
        INFILE(7)=0
        GOTO 800
430     INFILE(3)="DC"
        INFILE(4)="10"
        INFILE(5)="X."
        INFILE(6)="DB"
        IF (IFIVE.EQ.1) INFILE(6)="DF"
```

```
        INFILE(7)=0
        GOTO 800
440     INFILE(3)="F1"
        INFILE(4)="5."
        INFILE(5)="DB"
        IF (IFIVE.EQ.1) INFILE(5)="DF"
        INFILE(6)=0
        GOTO 800
450     INFILE(3)="MI"
        INFILE(4)="G2"
        INFILE(5)="1."
        INFILE(6)="DB"
        IF (IFIVE.EQ.1) INFILE(6)="DF"
        INFILE(7)=0
        GOTO 800
460     GOTO 410
470     GOTO 420
480     GOTO 430
490     GOTO 440
495     GOTO 450



800     TYPE"OPENING TARGET FILE...."

        IF (IPICK.GT.5) TYPE"ENTER FILENAME: "
        IF (IPICK.GT.5) READ(11,3) INFILE(1)
        CALL OPEN(2,INFILE,3,IER)
                IF (IER.NE.1) STOP"CANT OPEN FILE... SUCK IT UP!"

        IF (IFIVE.EQ.1) GOTO 850

        IF (IPICK.LT.6) GOTO 602
                TYPE"ENTER DIRECTION: "
                READ(11,601) IDIR
601             FORMAT(A2)
                WRITE(2,603) IDIR
603             FORMAT(1X,A2)
                ACCEPT"ENTER SPEED: ",MPS
                WRITE(2) MPS
                WRITE(2) RESOLU

602     WRITE(2)RESOLU
        WRITE(2) IXNUM
        DO 650 I=1,IXL
                IF (XVECT(I).GT.0) WRITE(2) XVECT(I)
650     CONTINUE
        WRITE(2) IYNUM
        DO 700 J=1,IYL
                IF (YVECT(J).GT.0) WRITE(2) YVECT(J)
700     CONTINUE
        CALL CLOSE(2)
```

```
        ACCEPT"ANOTHER TARGET? ? ?: ",IANS
        IF (IANS.EQ.1) GOTO 1
        GOTO 999

850     WRITE(2,890) RESOLU
        WRITE(2,895) IXNUM
        DO 860 I=1,IXL
                IF (XVECT(I).GT.0) WRITE(2,895) XVECT(I)
860     CONTINUE
        WRITE(2,895) IYNUM
        DO 870 J=1,IYL
                 IF (YVECT(J).GT.0) WRITE(2,895) YVECT(J)
870     CONTINUE
890     FORMAT(1X,F12.6)
895     FORMAT(1X,I5)
        CALL CLOSE(2)

999     TYPE"ALL DONE ..."
        END

C*************************************************************************
C
C       PROGRAM TGEN - TARGET GENERATOR         VERSION 2.0
C                                               22 JULY 1984
C                                               VERSION 1.0
C                                               17 JULY 1984
C
C
        INTEGER WIN, XPOS,YPOS,IXL,IYL,SCALE,XIN,YIN,PIX,NUMPOINTS,CNT
        INTEGER XSTART, XEND, ASTART,AEND,TEST
        REAL DIR,VERTLN,RESPIX,RTEMP

        INTEGER ICT(120),DISP(240),TARGETX(1000),TARGETY(1000),TARGETI(1000)
        INTEGER APIX(500)
        DIMENSION INFILE(20)

C
C
C       INITIALIZE OCKTEK DISPLAY
C

        IFIELD=0
        IX=0
        IXL=320
        IY=0
        IYL=240
        IENWV=15
        IENWC=15
        ICMAX=11
        CANGLE=0
        GRID=0
```

```
        CALL SINTRO (ICT,63K,IER)
        IF (IER.EQ.1) GOTO 51
                TYPE "INTRO ERROR CODE: ",IER
                STOP "UNABLE TO INITIALIZE"

51      CALL OPEN(3,"IACMON.XB",2,IER)
        IF (IER.NE.1) TYPE "WARNING: UNABLE TO ACCESS IACMON.XB"
        IF (IER.EQ.1) CALL LXB (ICT,3)

        CALL MPRUN (ICT,0,1)

C
C       OPEN TARGET DATA FILE
C
        TYPE
        TYPE
        TYPE "WELCOME TO TARGET GENERATOR"
        TYPE
        TYPE "ENTER TARGET FILENAME (NAME_DIR.DA): "
        READ(11,5) INFILE(1)
5       FORMAT(S20)
        CALL OPEN(2,INFILE,3,IER)
             IF (IER.NE.1) TYPE"CAN'T OPEN TARGET FILE"
             IF (IER.NE.1) STOP"ERROR CODE: ",IER
C
C       INITIALIZE VARIABLES
C
        ACCEPT"IS THIS A TEST: ",TEST
        WIN=240
        IXL=WIN
        XSTART=32
        XEND=WIN+XSTART-1
        IYL=1
C
C       GET INITIAL DATA
C
        TYPE
        ACCEPT"ENTER DIRECTION TARGET IS FACING (0-360): ",DIR
        DIR=DIR*0.017453
        TYPE
        ACCEPT"ENTER VERTICAL SCALE LENGTH OF DISPLAY (METERS): ",VERTLN
        RESPIX=VERTLN/240
        TYPE"FOR CURRENT DISPLAY, EACH PIXEL REPRESENTS",RESPIX,"MTRS"
        TYPE
        ACCEPT"DO YOU WANT TO (1)SPECIFY RESOLUTION OR (2)SCALE ?",IANS
        IF (IANS.EQ.2) GOTO 50
        TYPE
        ACCEPT"ENTER DESIRED PIXEL RESOLUTION: ",RDESRES
        RTEMP=RDESRES/RESPIX
        SCALE=IFIX(RTEMP)
        IDESRES=IFIX(SCALE*RESPIX)
        TYPE SCALE,IDESRES
```

```
        GOTO 60
C
C       TARGET SCALING LOOP
C
50      ACCEPT"ENTER SCALE REDUCTION (INTEGER): ",SCALE
        TYPE
60      IXL=IXL/SCALE
        TYPE"IXL: ",IXL
        DO 300 J=1,WIN,SCALE
                TYPE"LINE: ",J
                DO 200 I=XSTART,XEND,SCALE

C       CALCULATE X,Y COORDINATES OF REDUCED PIXEL

                        RTEMP=(I-XSTART)/SCALE+1
                        XPOS=IFIX(RTEMP)+1
                        RTEMP=J/SCALE
                        YPOS=IFIX(RTEMP)+1
                        PIX=0

                        XIN=SCALE
                        IF ((XIN+I).GT.XEND) XIN=(XEND-I)
                        YIN=SCALE
                        IF ((YIN+J).GT.YEND) YIN=(YEND-J)
                        IF (TEST.EQ.1) TYPE"READING IN VIDEO BLOCK: "
                        IF (TEST.EQ.1) TYPE"X:",I," Y:",J
                        IF (TEST.EQ.1) TYPE"XL:",XIN," YL:",YIN
                        CALL RVBLK(ICT,APIX,I,XIN,J,YIN)

C       ADD SCALE BY SCALE PIXELS INTO ONE VALUE, PIX

C       FIX SUMMING INDICES DUE TO FAULT IN OCTEK ROUTINES

                        ISQ=SCALE*SCALE
                        ASTART=2
                        AEND=ISQ+1
                        IF (TEST.EQ.1) TYPE"SUMMING ARRAY: ",ASTART,AEND
                        DO 100 K=ASTART,AEND
                                PIX=PIX+APIX(K)
100                     CONTINUE

C       SAVE PIXEL VALUE IN ARRAY

                RTEMP=PIX/(SCALE*SCALE)
                IF (TEST.EQ.1) TYPE"PIX(",I,J,"= ",RTEMP
                DISP(XPOS)=IFIX(RTEMP)
200             CONTINUE

C       WRITE ARRAY TO DISPLAY (ONE VIDEO LINE)
        CALL WVBLK(ICT,DISP,XSTART,IXL,YPOS,IYL)
300     CONTINUE
```

```
        RTEMP=WIN/SCALE
        WIN=IFIX(RTEMP)
        XEND=XSTART+WIN-1
        RESPIX=VERTLN/IXL
        TYPE
        TYPE"PRESENT SCALE: 1 PIXEL=",RESPIX,"METERS"
        ACCEPT"ANOTHER REDUCTION? ",IANS
        IF (IANS.EQ.1) GOTO 50
C
C       READ FINAL TARGET INTO ARRAY
C
        TYPE
        TYPE"READING TARGET DISPLAY"
        CNT=1
        IY=WIN-1
        DO 400 J=1,IY
                DO 400 I=XSTART,XEND
C       READ PIXEL I,J

                        IVAL=IRDPIX(ICT,I,J)
                        IF (IVAL.GT.14) GOTO 400
                                TARGETX(CNT)=I-XSTART+1
                                TARGETY(CNT)=J
                                TARGETI(CNT)=IVAL
                                CNT=CNT+1
400     CONTINUE

        NUMPOINTS=CNT-1
        IRANGE=XEND-XSTART
        TYPE "NUMBER OF POINTS: ",NUMPOINTS
        TYPE "WINDOW: ",IRANGE,WIN
C
C       OUTPUT DATA TO DISK
C
        TYPE
        TYPE"WRITING DATA TO DISK..."
        WRITE(2) NUMPOINTS
        WRITE(2) DIR
        WRITE(2) RESPIX
        WRITE(2) WIN
        WRITE(2) WIN

        DO 500 I=1,NUMPOINTS
                WRITE(2) TARGETX(I)
                WRITE(2) TARGETY(I)
                WRITE(2) TARGETI(I)
500     CONTINUE

        TYPE"ALL DONE !!!"
        END
```

```
C*************************************************************************
C
C       SUBROUTINE VECTOR-PREPROCESS          LANGUAGE:  FORTRAN V
C
C       CALLED BY: PROFILE.FR
C
C       FUNCTION:  REDUCES SAMPLED WINDOW DATA TO SINGLE X-Y
C                  VECTORS BY SUBTRACTING DC NOISE AND COMBINING
C                  WINDOWS BASED ON WINDOW CONTRAST
C
C*************************************************************************
      SUBROUTINE VPROC(MXW,IWPS,NSET,MXFR,NBND,NAVG,RN,RSEQ,
     1MXPNT,IPR,AVG,ICNT)

        INTEGER MXW      ;MAX NUMBER OF WINDOWS
        INTEGER MXFR     ;MAX NUMBER OF FRAMES
        INTEGER NBND    ;NUMBER OF WINDOWS
        INTEGER NAVG     ;NUMBER OF WINDOWS TO AVERAGE
        INTEGER IPR      ;PRINT SWITCH
        INTEGER ICNT     ;RETURN LENGTH OF AVG VECTOR
        INTEGER IWPS     ;WINDOWS PER SET
        INTEGER NSET     ;SETS OF WINDOWS

        INTEGER MXPNT(MXW)     ;TABLE OF WINDOW START POSITIONS
        REAL RN(MXW)           ;DC NOISE OF WINDOW
        REAL RSEQ(MXW,MXFR)    ;WINDOW DATA
        REAL AVG(MXFR)         ;RETURN VECTOR
        INTEGER ISORT(10)     ;INTERNAL WINDOW SORT ARRAY

        DO 400 I=1,IWPS

C       INITIALIZE VARIABLES

                ICNT=0
                ILSTBND=0
                ICURBND=I
                RMINNOISE=1000.0
                RMAXNOISE=0
                ISHIFT=0

                DO 50 J=1,MXFR
                        AVG(J)=0
50              CONTINUE

                DO 300 J=1,NSET

C       FIND NEXT LOWEST SNR WINDOW

                        RMINNOISE=1000
                        DO 100 K=I,NBND,IWPS
                           IF (RN(K).LE.RMAXNOISE) GOTO 100
                           IF (RN(K).GT.RMINNOISE) GOTO 100
```

```
                                        RMINNOISE=RN(K)
                                        ICURBND=K
100                     CONTINUE
                        RMAXNOISE=RMINNOISE ; LOW SNR NOW THRESHOLD

C       PRINT RESULTS

                        WRITE(10,110) ICURBND,RMINNOISE
                        IF (IPR.NE.1) GOTO 120
                        WRITE(12,110) ICURBND,RMINNOISE
110                     FORMAT(1X,"SELECTED WIN: ",I4," WITH DC
     1 NOISE: ",F12.6)

C       SHIFT BOUNDRIES TO ALIGN FOR AVERAGING

120                     IF (J.LT.2) GOTO 130 ; DONT SHIFT FIRST CHOICE
                        ISHIFT=MXPNT(ICURBND)-ILSTBND+ISHIFT
130                     ILSTBND=MXPNT(ICURBND)

C       AVERAGE SELECTED WINDOWS

                        DO 200 K=1,MXFR
                                INDEX=K+ISHIFT
                                IF (INDEX.LT.1) GOTO 200
                                IF (INDEX.GT.MXFR) GOTO 200
                                RTEMP=RSEQ(ICURBND,INDEX)
                                IF (RTEMP.GT.AVG(K)) AVG(K)=RTEMP
200                     CONTINUE
300             CONTINUE

C       PUT AVERAGED VECTOR IN FIRST WINDOW SET

                IRCNT=1
                DO 325 J=1,MXFR
                        IF (AVG(J).LE.0) GOTO 325
                                IF (IRCNT.GT.MXFR) GOTO 325
                                RSEQ(I,IRCNT)=AVG(J)
                                IRCNT=IRCNT+1
325             CONTINUE
                TYPE"NUMBER OF POINTS IN VECTOR( MINUS 1): ",IRCNT
                DO 350 J=IRCNT,MXFR
                        RSEQ(I,J)=0
350             CONTINUE
                WRITE(10,620) (RSEQ(I,J),J=1,MXFR)
400     CONTINUE

C    ORDER WINDOWS FOR RECOMBINATION BY CHOOSING LOWEST FIRST POINT

        RMAX=0
        RMIN=1000
        DO 410 I=1,IWPS
                ISORT(I)=1
```

```
410     CONTINUE
        DO 450 I=1,IWPS
                RMIN=1000
                DO 425 J=1,IWPS
                        IF (RSEQ(J,1).LE.RMAX) GOTO 425
                        IF (RSEQ(J,1).GT.RMIN) GOTO 425
                                RMIN=RSEQ(J,1)
                                ISORT(I)=J
425             CONTINUE
                RMAX=RMIN
450     CONTINUE
        DO 452 I=1,IWPS
                TYPE I,ISORT(I)
452     CONTINUE

C       COMBINE WINDOW SET TO FORM CONVOLUTED VECTOR

        ICNT=0
        RMAX=0
        DO 500 I=1,MXFR
                DO 500 J=1,IWPS
                        INDEX=(I-1)*IWPS+J
                        IF (INDEX.GT.MXFR) GOTO 500
                        AVG(INDEX)=RSEQ(ISORT(J),I)
                        IF (AVG(INDEX).GT.0) ICNT=ICNT+1
                        IF (AVG(INDEX).GT.RMAX) RMAX=AVG(INDEX)
500     CONTINUE

        WRITE(10,620) (AVG(I),I=1,ICNT)
        WRITE(12,610)
        WRITE(12,620) (AVG(I),I=1,ICNT)

        DO 600 I=1,MXFR
                AVG(I)=AVG(I)/RMAX
600     CONTINUE

        IF (IPR.NE.1) GOTO 999
                WRITE(12,610)
                WRITE(12,620) (AVG(I),I=1,ICNT)
610             FORMAT(1X,"     AVERAGE CONVOLUTION:")
620             FORMAT(1X,5F12.6)

999     RETURN
        END
```

Bibliography

1. Angyle, D.B., R.W. Christinsen, K. George, and T.J. Patterson, "Orbital Mosiac IR Image Simulation and Processing," Proceedings of the SPIE, 432: 82-89 (August 1983).

2. Benning, C. "Performance Considerations in Automatic Target Recognition Systems For Infrared Images." Proceedings of the SPIE, 359: 56-67 (August 1982).

3. Chow K., and J.P. Rode, "Signal Processing for Staring Infrared Images," Proceedings of the SPIE, 292: 204-209 (August 1981).

4. Castleman, Kenneth R. Digital Image Processing. New Jersey: Prentice-Hall Inc., 1979.

5. Darrell Chenoweth, Joeseph J. Kovar, and John Knect, "Automatic Classification of Infrared Ship Imagery," Proceedings of the SPIE, 292: 234-240 (August 1981).

6. Driscoll, Willaim and Martin Stern, "New Algorithm for Detection and Classification of Targets," Proceedings of the SPIE, 292: 249-255 (August 1981).

7. Evans, Robert C. "Background Suppression Techniques For Advanced Sensors," Presentation to the Background Measurements and Analysis Meeting. Lockheed Space and Missile Company, Palo Alto, California, Feb 1977.

8. Ho, C.Q. and R. Hu, "Perforamce Evaluation of Step Stare Sensor For Space-Based Air Vehicle Detection," Proceeding of the SPIE, 156: 30-35 (1978).

9. Hudson, Richard D. Infrared System Engineering. New York: John Wiley & Sons, Inc., 1969.

10. King, David A., PICBUF: A Virtual Memory System For Digital Picture Processing, Programmers Guide to Fortran Library used in the Signal Processing Laboratory, Air Force Institute of Technology, Wright-Patterson AFB, OH. (July 1984).

11. Lo, Chun Moo "Forward Looking Infrared Image Enhancement For a Generic Blob Target Detection System," Proceedings of the SPIE, 359: 210-213, (August 1982).

12. Mazaika, Paul K. "Jitter Induced Clutter In Staring Sensors Arising From Background Spatial Radiance Gradients," Optical Engineering, 21: 872-881 (September 1982).

13. Pohlman, R.T. "Staring Sensor Noise Sources," SPIE - Modern Utilization of Infrared Technology IV, 156: 138-147 (1976).

14. Pollock, David H. "Clutter Rejection For Infrared Surveillance Sensors," SPIE - Processing Of Images And Data From Optical Sensors, 292: 180-192 (1981).

15. Rauch, Herbert E. "Background Suppression and Tracking With a Starring Mosaic Sensor", Optical Engineering, 20: 103-110 (January 1981).

16. -------. "System Design For A Staring Mosiac Sensor," SPIE - Infrared Imaging System Technology, 226: 53-60 (1980).

17. Smith, M.C. and E.M. Winter, "Analysis Techniques for Two-Dimensional Infrared Data," Proceedings of the SPIE 156: 255-259, (1978).

18. Winter, E.M. "Performance Evaluation of Infrared Sensors and Associated Processing, " Proceedings of the SPIE, 95: 165-175 (1976).

19. Zvolanek, Budimir, "Autonomous Ship Classification By Invarient Moments," Proceedings of the SPIE, 292: 241-248, (August 1981).

Vita

Lieutenant James N.Tilley III was born on 30 April 1959 in Brooklyn, New York. He graduated from Central Dauphin High School in Harrisburg, Pennsylvania in 1977, and attended the United States Air Force Academy graaduating in 1981 with a Bachelor of Science in Electrical Engineering. He was then assigned to the Directorate of Technology, Space Division, Los Angeles as a project officer on two infrared satellite programs until entering the School of Engineering, Air Force Institute of Technology, in June 1983.

Permanent Address

3483 Bramlet Court
Clemmons, North Carolina 27012

REPORT DOCUMENTATION PAGE

REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved For Public Release; Distribution Unlimited
DECLASSIFICATION/DOWNGRADING SCHEDULE	
'ERFORMING ORGANIZATION REPORT NUMBER(S) AFIT/GE/ENG/84D-67	5. MONITORING ORGANIZATION REPORT NUMBER(S)
NAME OF PERFORMING ORGANIZATION School of Engineering 6b. OFFICE SYMBOL (If applicable) AFIT/ENG	7a. NAME OF MONITORING ORGANIZATION
ADDRESS (City, State and ZIP Code) Air Force Institute of Technology Wright-Patterson AFB, OH 45433	7b. ADDRESS (City, State and ZIP Code)
NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
ADDRESS (City, State and ZIP Code)	10. SOURCE OF FUNDING NOS. PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT NO.
TITLE (Include Security Classification) See Block #19	
PERSONAL AUTHOR(S) James N. Tilley III, B.S.E.E., 1Lt USAF	
L TYPE OF REPORT MS Thesis 13b. TIME COVERED FROM _____ TO _____	14. DATE OF REPORT (Yr., Mo., Day) 1984, December 15. PAGE COUNT 178
SUPPLEMENTARY NOTATION	

COSATI CODES		
IELD	GROUP	SUB. GR.
17		

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
Target Classification; Target Recognition
Infrared Surveillance; Pattern Recognition;
Signal Processing; Space Surveillance Systems

ABSTRACT (Continue on reverse if necessary and identify by block number)

THESIS TITLE: Low-Resolution Target Classification From A Staring Infrared Sensor

THESIS CHAIRMAN: Dr. M. Kabrisky
Professor of Electrical Engineering

Approved for public release: IAW AFR 190-17.
LYNN E. WOLAVER
Dean for Professional Development
Air Force Institute of Technology (AIC)
Wright-Patterson AFB OH 45433

DISTRIBUTION/AVAILABILITY OF ABSTRACT CLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS ☐	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
. NAME OF RESPONSIBLE INDIVIDUAL Dr. M. Kabrisky	22b TELEPHONE NUMBER (Include Area Code) 513-255-5276 22c. OFFICE SYMBOL AFIT/ENG

FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE.

A unique method to extract and classify aircraft targets from low-resolution staring infrared data is presented which uses sequences of sensor frames to obtain target signature samples. The classifier is based on a target feature description consisting of X and Y energy projections of the target. The target information is first extracted from sensor data through sampling windows consisting of contiguous rows and columns of detectors in the path of the expected target. The samples are then reduced to remove degradations from noise and sensor optics. The classifier assumes that approximate target position and velocity are available from a target detection and track subsystem. Tests were performed using a software sensor simulation that includes optical blur and sensor motion models. The classifier is shown through simulation to successfully extract and classify various transport and fighter class aircraft with sensor footprint resolutions of 11 to 45 meters.

END